AFRL-IF-RS-TR-2006-68
Final Technical Report
February 2006

REGISTRATION AND TRACKING REPORT

University of Notre Dame

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** FEBRUARY 2006	**3. REPORT TYPE AND DATES COVERED** Final Apr 04 – Dec 05

4. TITLE AND SUBTITLE
REGISTRATION AND TRACKING REPORT

5. FUNDING NUMBERS
C - FA8750-04-1-0063
PE - 62702F
PR - 558B
TA - BA
WU - 01

6. AUTHOR(S)
Robert Stevenson,
Patrick Flynn and
Kevin Bowyer

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Notre Dame
Office Of Research
511 Main Bldg
Notre Dame Indiana 46556-5602

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/IFEC
525 Brooks Road
Rome New York 13441-4505

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2006-68

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: Jonathan Gregory/IFEC/(315) 330-4294/ Jonathan.Gregory@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 Words)*
Analysis of algorithms and methods to register video data acquired from multiple sensors and the ability to track items/objects across and along paths through multiple sensors areas of coverage.

14. SUBJECT TERMS
Multisensor Registration Tracking Classification Objects

15. NUMBER OF PAGES
106

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

List of Figures

List of Tables

Executive Summary

Force protection, intelligence gathering, and targeting systems all use sensors to collect necessary information about the environment so that intelligent decisions can be made about proper course of action. In most systems a single sensor type (single mode) is utilized. An investigation into a multimodal video system for object tracking was conducted. Problems in sensor registration and tracking arise in this situation and techniques were examined to address these issues.

The multisensor registration problem refers to the need to correlate the spatial locations between multiple sensors. When multiple types of sensors collect information about a scene they often dont have the same point of view or same resolution and may collect data at different points in time. The resulting task is to determine how data from one data set relates to data in another. The goal is to determine the proper geometric transformation of the data so that spatial correspondence between the image data sets can be established.

The availability of registered multimode sensing systems enables a variety of interesting applications of significance to civilian security and military domains. Assessment of threats often employs evidence gathered over time. Hence, the ability to detect and track objects of potential interest (vehicles, ordnance, human bodies individually or in crowds, etc.) as they move through a sensor's field of regard is a necessary component of such evidence aggregation techniques. Tracking systems must incorporate the ability to disambiguate tracks after occlusion, to reacquire and retrack automatically, and to summarize object dynamics at various levels. Object recognition is also a component of such systems in that it provides a mechanism for reacquisition of a previously tracked object.

Algorithms were developed and tested for the multimodal registration and integration/tracking

problems. All development work was done on a Pentium workstation in C++. All prototype software was provided.

Chapter 1

Object Tracking Survey

Object tracking is a crucial research topic within the area of computer vision. The proliferation of high powered computers and the increasing need for automated surveillance systems have generated a great deal of interest in object tracking algorithms. Some of the tasks that use object tracking are:

- *Crowd flux statistics.* Real time gathering of traffic statistics to direct traffic flow. Using techniques for human tracking, surveillance systems can automatically compute the flux of people at important public areas to assist in traffic management.

- *Access Control.* Analyzing the walking gait to decide the visitor's qualification for entry. In security-sensitive areas, only people with a special identity are allowed to enter. When somebody is about to enter, the system could track and analyze the walking gait to decide whether the visitor can be cleared for entry.

- *Anomalous behaviors detection.* Real time monitoring a scene to detect suspicious activities or unlikely events. Visual surveillance systems set in parking lots and supermarkets could analyze abnormal behaviors indicative of crime.

- *Video indexing.* Automatic annotation and retrieval of videos for multimedia databases.
- *Vehicle navigation.* Navigating vehicles with path planning and obstacle avoidance capabilities.

Tracking can be defined as the problem of estimating the trajectory of an object as it moves around a scene. Object tracking, in general, is a challenging problem. Difficulties in tracking objects can arise due to abrupt object motion, loss of information caused by projection of the 3D world on a 2D image, changing appearance patterns of both the object and the scene, non-rigid object structures, occlusions, noise in image and camera motion.

Usually, the tracking problem is simplified by imposing constraints on the motion and appearance of the objects, and by using a priori knowledge. Most of the tracking algorithms assume the object motion to be smooth. In certain scenarios, one can further constrain the object motion to be of constant velocity or constant acceleration based on a priori information. Prior knowledge about the number, size and appearance of the objects can also be used to simplify the problem.

The major difference among the existing tracking algorithms lies in the way they represent target objects; the way they model the motion; the appearance and the shape of the object and the image features they use. One primary goal of this survey is to group tracking methods into appropriate categories and provide comprehensive descriptions of representative methods in each category.

The remainder of this chapter is organized as follows: Section 1.1 describes object shape representations, appearance representations and image features which are usually used in object tracking. In section 1.2, we summarize the general strategies for object detection in a scene. We categorize and describe the existing tracking methods in section 1.3. Also, we explain their strengths and weaknesses at the end of each category. The important issues and future development are discussed in section 1.4.

1.1 Object Representation/Image Feature For Tracking

In this section, we will first describe object and appearance representations for tracking and then address the feature selection issues for tracking.

1.1.1 Object Representation

Commonly used object representations for tracking can be categorized as follows:

- *Centroid points.* The object can be effectively represented, in some contexts, by centroid point. In general, this shape representation is suitable only when the tracking objects occupy very small regions in an image.

- *Geometric shapes.* The object shape is represented by a rectangle, ellipse, etc. Object motion in this shape representation is usually modelled by translation, affine or projective transformation. The primitive geometric shapes are suitable for both rigid objects and non-rigid objects.

- *Contour.* The contour representation defines the boundary of the object and is suitable for representing complex non-rigid objects.

- *Articulated shape models.* Articulated objects are composed of object parts that are held together by joints. The relationships between the parts are governed by the model parameters, e.g., joint angle. To represent the constituent parts of the articulated model, one can use lines, ellipses or cylinders.

1.1.2 Appearance Representation

There are a number of ways to represent the appearance features of objects. Shape representations can also be combined with the appearance representations for tracking. Some common appearance representations in the context of tracking algorithms are given below:

- *Probability densities based appearance model.* The object appearance can be effectively modelled by the feature probability density. The probability density estimates of the object appearance can either be parametric, e.g., Gaussian [1] or Gaussian Mixture model [2], [3], or non-parametric, e.g., Parzen windows [4], [10] or histograms [5], [11]. The PDF of object appearance features, e.g., color, can be computed from the image regions specified by the shape models.

- *Templates.* Templates are formed using simple geometric shapes or silhouettes [12]. It carries both spatial and appearance information, but only encodes the appearance of the objects from one view.

- *Multiview models.* This representation complements the previous one by encoding different views of an object. This model can be achieved by training a classifiers, e.g., support vector machines [13], or by representing all the views of an object using eigenspace decomposition [14]. Obviously, the appearance information of all views of the tracking objects has to be available ahead of time.

Object representations are usually chosen according to the tracking application. For example, if the tracking objects appear very small, centroid point representation is enough [15], [16]. For objects that have complex shapes, like a human, a contour or a silhouette based representation is appropriate [6], [7], [17].

1.1.3 Good Features for Tracking

The most desirable property of a visual feature for tracking purpose is its uniqueness, so that the objects can easily be distinguished in the feature space. Many tracking algorithms use a combination of features to achieve uniqueness. Most commonly used features include color, texture, edge and optical flow. Color is primarily generated by the reflectance of the light from the object surface. The texture feature measures the variation of the color. Edges are generated by the strong changes in image intensities. Optical flow represents a dense field of displacement vectors which defines the translation of each pixel in a region. Selecting the right features is closely related to the object representation. For example, color is used as a feature for histogram based appearance representations [11], while, for contour based representation, edges and optical flows are usually used as features [18].

1.2 Object Detection

Nearly all visual tracking systems start with object detection. Object detection aims at segmenting regions corresponding to moving objects from the rest of an image. Subsequent processes such as tracking and behavior recognition are greatly dependent on it. The process of object detection usually involves background modelling, object segmentation, which intersect each other during processing

1.2.1 Background Modelling and Subtraction

The basic approach is to maintain a model of the static background and compare the current frame with the background. A moving object is detected by finding the significant change in the difference image. Further processing is needed for the pixels regions that are labelled as target candidates. A connected component algorithm is an important next step and often utilized to obtain connected regions which are corresponding to the objects. Wren et al. [1] marks the popularity of background subtraction methods. In their paper, Wren et al. proposed to model the color of each pixel of a stationary background with a single 3D (YUV color space) Gaussian, $I(x,y) \rightarrow N(\mu(x,y), \sigma(x,y))$ to learn gradual changes over time. The mean $\mu(x,y)$ and the covariance $\sigma(x,y)$ are learned in the first several consecutive frames. Once the background model is captured, for every pixel (x, y) in the input frame, the Probability of its color coming from $N(\mu(x, y), \sigma(x, y))$ is computed, and the pixels that deviate from the background model are labelled as the foreground pixels. In many situations, however, a single Gaussian is not enough [20], since multiple colors may be observed at a certain location due to repetitive object motion, shadows or reflectance from multiple surfaces. A substantial improvement in background modelling is achieved by using multi-modal statistical models to describe the per pixel background color. Stauffer and Grimson [19] use a mixture of Gaussians to model the pixel color. In their method, a pixel in the current frame is checked against the background model by comparing it with every Gaussian in the mixture model until a matching Gaussian is found. If a match is found, the mean and variance of the matched Gaussian are updated, otherwise a new Gaussian with the mean equal to the current pixel color and some initial high variance is introduced into the mixture model. Each pixel is classified as background or foreground based on what process the matched distribution represents. Instead of using parametric model to representing pixel process, Elgammal et. al [85] introduced a novel background model and subtraction technique based on nonparametric statistical modelling. The model keeps a sample of intensity values for each pixel in the image and uses this sample to estimate the probability density function of the pixel intensity. The density function is estimated using kernel density estimation technique. Since this approach is quite general and might be seen as a generalization of Strauffer and Grimson [19]'s work, the model can approximate any distribution for the pixel intensity without any assumptions about the underlying distribution shape. The model can handle situations where the background of the scene is cluttered and not completely static but contains small motions that are due to moving tree branches and bushes. The model is updated continuously and therefore adapts to changes in the scene background. Furthermore, the approach runs in real-time.

Alternatively, the variations of the pixel intensity can be represented as discrete states i.e., different events in the environment (e.g., for tracking cars on a highway, image pixels can be in the background state, the foreground (car) state or the shadow state). Rittscher et al. [20] use

Hidden Markov Model to classify small blocks of an image as belonging to one of these three states. Stenger et al. [21] use Hidden Markov Model, in the context of detecting light on and off events in a room, for the background subtraction.

Oliver et al. [22] adaptively build an eigenspace that models the background. This eigenspace model describes the range of appearances (e.g., lighting variations over the day, weather variations, etc.) that have been observed. The eigenspace model is formed by takeing a sample of N images and computing both the mean μ_b background image and its covariance matrix C_b. This covariance matrix can be diagonalized via an eigenvalue decomposition. In order to reduce the dimensionality of the space, the author keeps only M most important eigenvectors (eigenbackgrounds) that encompass all possible illuminations in the field of view. Thus, this approach is less sensitive to illumination. The foreground objects don't have a significant contribution to this model, hence the portions of an image containing a moving object cannot be well described by this eigenspace model, so the object can be detected by projecting the current image to the Eigenspace and finding the difference between the reconstructed and actual images. This approach is sensitive to background variations. Monnet et al. [23], and Zhong et al. [24] addressed the limitation of a static background model recently. Both of these methods are able to deal with time varying background, for example, storms, moving clouds. These methods model the image regions as autoregressive moving average processes to learn and predict the motion patterns in a scene.

Background subtraction techniques are widely used in fixed camera tracking applications because recent advances are able to model the changing illumination, noise, and the periodic motion of the background regions and therefore can accurately detect objects in a variety of circumstances. Moreover, these methods are computationally efficient. In practice, background subtraction provides incomplete object regions in many instances, i.e., the objects maybe split into several regions or there may be holes inside the object since there are no guarantees that the object features will be different from the background features. The most important limitation of background subtraction is the requirement for stationary cameras. Camera motion usually distorts the background models. An extension of these methods to mobile camera can be obtained by regenerating background models for a small temporal window [27][28][29]. Note that these extensions require that the motion in successive frames is small.

1.2.2 Video Segmentation

Video segmentation can be defined as the segmentation of image sequences into differently moving objects (or, more correctly, their projections in the image). In this section previous work on video segmentation is reviewed.

Comaniciu et. al [30] propose a general nonparametric estimator of density to analyze a complex multimodal feature space and delineate arbitrarily shaped clusters in it. The basic computational module of the technique is an old pattern recognition procedure called the mean shift. Given an image, the algorithm is initialized with a large number of hypothesized cluster centers randomly chosen from the data. Each cluster center is then moved to the mean of the data lying inside the multi-dimensional ellipsoid centered on the cluster center. The vector defined by the old and the new cluster centers is called the mean shift vector. The mean shift vector is computed iteratively until the cluster centers do not change their positions. This segmentation technique requires fine tuning of various parameters to obtain better segmentation. Another limitation is its high computational cost.

Graph cut based video segmentation is another very interesting approach . The basic idea of this approach is: each image pixel is viewed as a vertex of a graph, the similarity between two pixels is viewed as the weight of the edge of these two vertices, and segmentation is achieved by cutting edges in the graph to form a good set of connected components. The weights of the within-component edges will be large compared to the across-component edges. As a result, image segmentation problem becomes a graph cut problem. Zhi et. al [29] propose the normalized cut to overcome the over-segmentation problem. In their approach, the cut not only depends on the sum of the edge weights in the cut, but also on the total connection weight of nodes in each partition to all nodes of the graph. For image based segmentation, the weights between the nodes are defined by the product of the color similarity and the spatial proximity. One limitation in this graph-cut method is that the processing and memory requirements can be expensive for large images. However, compared to mean shift segmentation, it requires fewer manually selected parameters.

Optical flow based methods solve video segmentation problem by using characteristics of flow vectors of moving objects over time to detect moving regions in an image sequence[78]. It computes an approximation to a projection of the 3D velocities of surface points onto the image surface, from spatial and temporal patterns of image intensity. Once computed, the measurements of image velocity can be used for segmentation of video sequence. However, the measurement has to be accurate and dense, providing a close approximation to the 2D motion field, in order to perform correct segmentation.

Active contour is another way to achieve image segmentation. It evolves a closed contour to the object's boundary so that the contour tightly encloses the object region. Evolution is directed by energy functionals which define the fitness of the contour to the hypothesized object region. Contour energy functionals have the following form:

$$E(\Gamma) = \int E_{int}(v) + E_{im}(v) + E_{ext}(v)ds$$

where s is the arc length of the contour Γ, E_{ext} includes regularization constraints, E_{im} includes appearance based energy and E_{ext} specifies additional constraints. E_{int} usually includes a curvature term, first order or second order continuity terms to find the shortest contour. E_{im} is computed from the image gradient which is evaluated around the contour [35, 36], or the color [1], [38] and texture [2] information which is evaluated inside and outside the object region. Different variations of the energy functional also exist [35], [36].

An important issue in contour based methods is the contour initialization. A common approach is to place the contour outside the object region and shrink until the object boundary is encountered [32]. This constraint is relaxed in region based methods, such that the contour can be initialized either inside or outside the object, so that the contour can either expand or shrink respectively, to fit the object boundary. However, these approaches usually require prior object or background knowledge. Using multiple frames or a reference frame, initialization can be performed without building region priors. For instance in [39], the authors used background subtraction to initialize the contour. Besides the selection of the energy functional and the initialization, another important issue is to select the right contour representation. The contour is represented either explicitly or implicitly. In the explicit representation, the relation between the control points is defined by the spline equations. In the implicitly representation, the contour is represented on a spatial grid which encodes the signed distances of the grids from the contour with opposite signs for the object and the background regions. The contour is evolved by modifying the grid values. Both of these representations have their advantages and disadvantages. For instance, the most important advantage of the implicit representation over the explicit representation is its flexibility to allow topology changes. However, due to representing the contour on a grid, contour evolution using implicit representation is computationally more expensive than the explicit representation.

Table 1.1: Object Detection Categories

Categories	Representative Work
Background Modelling And Subtraction	Gaussian Mixture Model [19]
	Eigenbackground [22]
	Nonparametric Model [10]
	Dynamic texture background [23]
Video Segmentation Techniques	Mean Shift [80]
	Active Contours [36]
	Optical Flow [79]

1.3 Object Tracking

Object tracking can be defined as locating the object and finding the region it encompasses in the image at every time instant. These two operations can be performed separately or jointly. For instance, first the regions of interest can be found by a detection algorithm and then they can be corresponded with the objects previously observed. Alternatively, the regions corresponding to the objects can be estimated iteratively, given the previous location and some similarity criteria. In either case, it is necessary to represent the region that the object encompasses using one of the shape models described in Section 1.1. The modelling of object motion directly depends on its representation. For example, if an object is represented as a point then only a translational model completely defines its motion. If a geometric shape representation is used then parametric motion models like affine or projective transformations can be used. These representations can approximate the motion of rigid objects in the scene. For non-rigid objects, contours are the most descriptive representation and both parametric and non-parametric models can be used to specify their motion.

The tracking approaches can be broadly classified into three categories: **Tracking by point correspondence**, **by matching geometric regions**, and **by contour evolution.**

These methods usually use a combination of shape and appearance to model the object. For example an object can be represented by a rectangular template, or by an elliptical shape and an associated histogram. Objects are tracked by estimating the parametric transformation of the shape model in consecutive frames. Object detection is required only in the first frame for these methods.

Tracking is performed by estimating the object contour, given an initial contour from the previous frame. This essentially can be considered as segmentation of each frame using priors obtained from previous images.

Table 1.2: Object Tracking Categories

Categories	Representative Work
Tracking by Point Correspondence	Salari tracker [47]
	Veenman tracker [15]
	Kalman Filter [51]
	JPDAF[67], PMHT [59]
Tracking by matching geometric regions	Mean Shift [11]
	KLT [42]
	Layering [67]
	Eigentracking [14]
	SVM tracker [13]
Tracking by contour evolution	Condensation [71]
	JPDAF+HMM [57]
	Variational methods [77]
	Heuristic methods [36]

1.3.1 Tracking by Point Correspondence

Tracking can be formulated as correspondence of detected objects represented by points across frames. Point correspondence is a complicated problem, especially in the presence of occlusions, misdetections, entries and exits of objects. Overall, the object correspondence approaches can be divided into two broad categories, namely deterministic methods and statistical methods. The deterministic methods use ”qualitative motion heuristics” [15] to constrain the correspondence problem. The correspondence solution is found through a combinatorial optimization scheme. On the other hand, probabilistic methods explicitly take the object measurement and model uncertainties into account to establish correspondence. The object state consists of object kinematics such as position, velocity and acceleration. A Maximum a Posteriori (MAP) estimate of the object state is calculated at each time instant. Below, we describe the tracking approaches belonging to these two subcategories in detail.

Deterministic Methods for Correspondence

The deterministic methods for correspondence define a cost of associating each object in frame $t-1$ to a single object in frame t using a set of motion constraints. Minimization of the correspondence cost (e.g., proximity, Maximum velocity, Small velocity change, etc.) is formulated as a combinatorial optimization problem. A solution, which consists of one-to-one correspondences among all possible associations, can be obtained by optimal assignment methods, e.g., the Hungarian algorithm or greedy search methods. Sethi et. al [46] solve the correspondence by a greedy approach based on the proximity and rigidity constraints. Their algorithm considers two consecutive frames, and is initialized by the nearest neighbor criterion. The correspondences are exchanged iteratively to minimize the cost. A modified version of the same algorithm is also analyzed, which computes the correspondences in the backward direction (from the last frame to the first frame) in addition to the forward direction. This method cannot handle occlusions, entries or exits. Salari et. al [47] handle these problems by first establishing correspondence for the detected points and then extending the tracking of the missing objects by adding a number of hypothetical points. Rangarajan et. al [48] propose a greedy approach, which is constrained by proximal uniformity. Initial correspondences are obtained by computing the gradient based optical flow vector in the first two frames. The method does not address entry and exit of objects. If the number of detected points decreases occlusion or misdetection is assumed. Occlusion is handled by establishing the correspondence for the detected objects in the current frame. For the remaining objects, the position is predicted based on a constant velocity assumption. In the work by Intille et al. [49], which uses a slightly modified version of [48] for matching centroids of objects, the objects are detected using background subtraction. The authors explicitly handle the change in number of objects by examining specific regions in the image, for example, a door, to detect entries/exits before computing the correspondence. Veenman et al. [15] extend the work of [49, 51] by using common motion constraints for correspondence. The algorithm is initialized by generating the initial tracks using a two-pass algorithm, and the cost function is minimized by the Hungarian assignment algorithm in two consecutive frames. This approach can handle occlusion and misdetection errors, however, it is assumed that the number of objects is the same throughout the sequence, i.e., no object enters or exits. Shafique et. al [16] propose a multi-frame approach to preserve temporal coherency of the speed and position. They represent the correspondence as a graph theoretic problem. Multiple frame correspondence relates to finding the best unique path for each point. For mis-detected or occluded objects, the path will consist of missing positions in corresponding frames. The directed graph, which is generated using the points in k frames, is converted to a bipartite graph by splitting each node (object) into two (+ and -) nodes and representing directed edges as undirected edges from + to - nodes. The correspondence is then established by a greedy algorithm.

Statistical Methods for Correspondence

The object motions undergo random perturbations. Furthermore, measurements obtained from sensors invariably contain noise. The statistical correspondence methods use the state space approach to model the object properties such as position, velocity and acceleration. These methods treat the tracking problem as inferring the object state by taking the measurement and the model uncertainties into account. Measurements usually consist of the object position in the image, which is obtained by a detection mechanism. Below, we will discuss the probabilistic state estimation methods in the context of point tracking, however it should be noted that these methods can be used in general to estimate the state of any time varying system.

Consider a moving object in the scene. The information representing the object, e.g., location is defined by a sequence of states $X^t : 1, 2, ...$ The change in state over time is governed by the dynamic equation,

$$X^t = f(X^{t-1}) + W^t$$

The relationship between the measurement and the state is specified by the measurement equation:

$$Z^t = h(X^t, N^t)$$

The objective of tracking is to estimate the state X^t given all the measurements up to that moment, or equivalently, to construct the probability density function $p(X^t|Z^{1,\ldots t})$. The optimal solution is provided by the recursive Bayesian filter which solves the problem in two steps. The prediction step uses the dynamic equation and the already computed PDF of the state at previous time to derive the prior PDF of the current state. Then, the correction step employs the Bayes formula to computer the posterior PDF $p(X^t|Z^{1,\ldots t})$ using the likelihood function $p(Z^t|X^t)$. If there is only one object in the scene, then the state can be simply estimated by the two steps defined hitherto. On the other hand, if there are multiple objects in the scene then measurements need to be associated with the corresponding object states. We discuss the two cases in the following

For the single object case, if the dynamic and measurement equations are both linear, and the noise have a Gaussian distribution, then the optimal state estimate is given by the Kalman Filter.

In the general case, state estimation can be performed by using particle filters.

The Kalman filter addresses the general problem of trying to estimate the state of a discrete-time controlled process that is governed by the linear stochastic difference equation. It assumed the states of the system to be distributed as a Gaussian.

The state update model of the Kalman filter is:

$$x_{k+1} = A_k x_k + B_k u_k + w_k$$

The observation model that relates the measurement to the current state is:

$$z_k = H_k x_k + v_k$$

where,w_k and v_k represent the process and measurement noise (respectively). They are assumed to be independent (of each other), white, and with normal probability distributions

$$P(w) \rightarrow N(0, Q)$$

$$P(v) \rightarrow N(0, R)$$

The Kalman filter algorithm proceeds in the following way:

1) Predict the current state given the previous information:

$$y_k = Ax_{t-1} + Bu_{t-1}$$

2) Predict error of the predicted state:

$$E_t = AP_{t-1}A^T + Q$$

3) Estimate correction gain between actual and predicted observations:

$$K_t = E_t H^T (HE_t H^T + R)^{-1}$$

4) Estimate new state given prediction and correction from Observations:

$$x_t = y_t + K_t(z_t - Hy_t)$$

5) Estimate error of the estimated state:

$$P_t = (I - K_t H)E_t$$

The Kalman filter has been extensively used in the vision community for tracking. Broida and Chellappa [51] used Kalman filters to track points in noisy images. In stereo camera based object tracking, Beymer and Konolige [52] use it for predicting the object's position and speed in x-z dimensions. Rosales and Sclaroff [56] use the extended Kalman filter to estimate the 3D trajectory of an object from 2D motion. A Matlab toolbox for Kalman filtering is available at [54].

One limitation of the Kalman filter is the assumption that the state variables are normally distributed (Gaussian). Thus modeling state variables that do not have Gaussian distributions with the Kalman filters will result in poor state estimations. This limitation can be overcome by using particle filtering [55].

In particle filtering, the conditional state density $p(X_t|Z_t)$ at time t is represented by a set of samples $\{s_t^n : n = 1, ..., N\}$ particles with weights $\pi_t^{(n)}$ (sampling probability). The weights define the importance of a sample. The procedure of the particle filtering algorithm can be stated as follows:

1) Select N random samples, by generating a uniformly distributed random number, from the sample set $\{s_t^n : n = 1, ..., N\}$

2) Predict a new sample $s_t^{(n)}$ from the a distribution centered at $s_{t-1}^{(n)}$

3) Correct the new sample $s_t^{(n)}$using dynamic models and Kalman filter.

4) Update weights $\pi_t^{(n)}$

Note that the Kalman and particle filter described above assume a single measurement at each time instant, i.e., the state of single object is estimated. Tracking multiple objects requires a joint solution of data association and state estimation problems.

For the multiple objects case, we need to deterministically associate the most likely measurement for a particular object to that object's state, i.e., the correspondence problem needs to be solved before these filters can be applied. The simplest method to perform correspondence is to use the nearest neighbor approach. However, if the objects are close to each other, then there is always a chance that the correspondence is incorrect. An incorrectly associated measurement can cause the filter to fail to converge.

There exist several statistical data association techniques to tackle this problem. Among them, Joint Probability Data Association Filtering and Multiple Hypothesis Tracking are two most widely used techniques for data association. Joint Probability Data Association Filtering is used by Chang et. al [56] to perform 3D structure reconstruction from a video sequence. Rasmussen et. al[57] use a constrained Joint Probability Data Association Filtering filter to track regions. The major limitation of the Joint Probability Data Association Filtering algorithm is its inability to handle new objects entering the field of view or already tracked objects exiting the field of view. Since the Joint Probability Data Association Filtering algorithm performs data association of a fixed number of objects being tracked over two frames, serious errors can arise if there is a change in the number of objects. The Multiple Hypothesis Tracking algorithm, on the other hand, does not have this shortcoming: If motion correspondence is established using only two frames, there is always a finite chance of an incorrect correspondence. Better tracking results can be obtained if the correspondence decision is deferred until several frames have been examined. The Multiple Hypothesis Tracking algorithm maintains several correspondence hypotheses for each object at each time frame [59]. The final track of the object is the most likely set of correspondences over the time period of its observation. The algorithm has the ability to create new tracks for objects entering the field of view and terminate tracks for objects exiting the field of view. It can also handle occlusions, i.e., continuation of a track even if some of the measurements from an object are missing.

Multiple Hypothesis Tracking is an iterative algorithm. An iteration begins with a set of current track hypotheses. Each hypothesis is a collection of disjoint tracks. For each hypothesis, prediction of each object's position is made for the next frame, and then the predictions are compared with actual measurements by evaluating a distance measure. A set of correspondences (associations) are established for each hypothesis based on the distance measure, which introduces new hypotheses for the next iteration. Each new hypothesis represents a new set of tracks based on the current measurements. Note that each measurement can belong to a new object entering the field of view, a previously tracked object, or a spurious measurement. Moreover, a measurement may not be assigned to an object because the object may have exited the field of view, or a measurement corresponding to an object may not be obtained. The latter happens when the object is occluded or it is not detected due to noise.

Note that Multiple Hypothesis Tracking makes associations in a deterministic sense and exhaustively enumerates all possible associations. To reduce the computational load, Streit et. al[59] propose a probabilistic Multiple Hypothesis Tracking in which the associations are considered to be statistically independent random variables. Thus there is no requirement for exhaustive enumeration of associations. Recently, particle filters that handle multiple measurements to track multiple objects have been proposed by Hue et al. [60]. In this method the data association is handled in similar manner to Probability Multiple Hypothesis Tracking, but the state estimation is achieved through particle filters.

The Multiple Hypothesis Tracking algorithm is computationally exponential both in memory and time. To overcome this limitation, Cox et. al [61] use Murty's [62] algorithm to determine the k-best hypotheses in polynomial time for tracking interest-points. Cham et. al [63] use the multiple hypothesis framework to track the complete human body.

Discussion and Evaluation

In order to tackle noisy or missing observations, deterministic point trackers use heuristic constraints, i.e., common motion [15] or proximal uniformity [48]. These methods usually use a greedy approach to establish the correspondence by minimizing a cost function. Thus the solution is not necessarily optimal. Statistical point tracking methods explicitly model the measurement and model uncertainties for tracking. These uncertainties are usually assumed to be in the form of normally distributed noise. However, the assumption that measurements are normally distributed around their predicted position may not hold. Moreover, in many cases the noise parameters are not known. In the case of valid assumptions on distributions and noise, Kalman filters [64] and Multiple Hypothesis Tracking [59] give optimal solutions.

Table 1.3: Object Tracking Categories

	Obj No.	Entry	Exit	Occlusion	Optimal
Salari [47]	Multi	Yes	Yes	Yes	No
Veenman [15]	Multi	No	No	Yes	Yes
Multiframe [16]	Multi	Yes	Yes	Yes	No
Kalman [64]	Single	No	No	No	Yes
JPDAF [64]	Multi	No	No	No	No
MHT [61]	Multi	Yes	Yes	Yes	Yes

In Table 1.3, we provide a qualitative comparison of point correspondence tracking algorithm based on their ability to deal with entries of new objects, exits of objects, missing observations (occlusion) and to provide optimal solutions for correspondences.

1.3.2 Tracking by Matching Geometric Regions

Given the object in the first frame, these trackers compute the parametric motion, e.g., translation, translation and rotation, and affine transformation, of the object in subsequent frames. These algorithms differ in terms of the appearance representation used, the number of objects tracked and the method used to estimate the motion parameters. We divide these tracking methods into two subcategories based on the appearance representation used, namely templates and density based appearance models, and multi-view appearance models.

Tracking by Template and Density based Appearance Models

Templates and density based appearance models have been widely used in the area of tracking. In general, these trackers model the object and its motion independently of the other objects in the scene. However, in the case of multi-object tracking, the independence assumption may not be valid. We divide the trackers in this category into two sub-categories based on whether the objects are tracked individually or jointly.

For tracking single objects case, The most common approach in this category is ”template matching”. Template matching is a brute force method of searching the image, for a region similar to the object template.The position of the template in the current image is computed by a similarity measure, e.g., cross correlation. Usually image intensities or color features are used to form the templates. Since image intensity is very sensitive to illumination changes, image gradients [65] can also be used as features. A limitation of template matching is high computation cost due to the brute force search. To reduce the computational cost, researchers usually limit the object search to the vicinity of its previous position.

Note that, instead of templates, other object representations can also be used for tracking, for instance, color histograms or mixture models can be computed by using the pixels inside the rectangular or ellipsoidal regions. Fieguth et. al [12] generate object models by finding the mean color of the pixels inside the rectangular object region. To reduce computational complexity, they search for the object in eight neighboring locations. The similarity between the object model, M, and the hypothesized position, H, is computed by evaluating the ratio between the color means computed from M and H. The position that provides the highest ratio is selected as the current object location.

Comaniciu et. al [11] use a weighted histogram computed from an elliptical region to represent the object. Instead of performing a brute force search for locating the object, they use the mean shift procedure. The mean shift tracker maximizes the appearance similarity iteratively by comparing the histograms of the object, Q, and the window around the hypothesized object location, P. At each iteration, the mean shift vector is computed such that the histogram similarity is increased. This process is repeated until convergence is achieved. For histogram generation, the authors use a weighting scheme defined by a spatial kernel which gives higher weights to the pixels closer to the object center.

An obvious advantage of the mean shift tracker over the standard template matching is the elimination of a brute force search, and the computation of the translation of the object patch in a small number of iterations. However, mean shift tracking requires that a portion of the object is inside the elliptical region upon initialization.

Shi et. al[42] propose the KLT tracker based on the principle that only good features can be tracked well. Their approach consists of two steps. The first step finds the translation of an interest point and the second step monitors the quality of each interest point. Given a set of interest points computed using the KLT detector, the translation of the patch centered on the point is iteratively computed. Once the new location of the interest point is obtained, the authors compute the affine transformation between the corresponding patches in consecutive frames. If the sum of square differences between the current patch and the projected patch is small, they continue tracking the feature, otherwise the feature is eliminated.

Jepson et al. [66] propose an object tracker that tracks an object as a three component mixture, consisting of the stable appearance features, transient features and noise process. The stable component identifies the most reliable appearance for motion estimation, i.e., the regions of the object whose appearance does not quickly change over time. The transient component identifies the quickly changing pixels. The noise component handles the outliers in the object appearance, which arise due to noise. An online version of the EM algorithm is used to learn the parameters of this three component mixture. The authors use the phase of the steerable filter responses as features for appearance representation. The object shape is represented by an ellipse. The motion of the object is calculated in terms of warping the tracked region from one frame to the next one. The warping transformation consists of translation, rotation and scale parameters. A weighted combination of the stable and transient components is used to determine the warping parameters. The advantage of learning stable and transient features is that one can give more weight to stable features for tracking, for example, if the face of a person who is talking is being tracked, then the forehead or nose region can provide a better match to the face in the next frame as opposed to the mouth of the person.

For tracking multiple objects case, modelling objects individually does not take into account the interaction between multiple objects and between objects and background during the course of tracking. An example interaction between objects can be one object partially or completely occluding the other. The tracking methods given below model the complete image, i.e., the background and all moving objects are explicitly tracked.

Tao et al. [67] propose an object tracking method based on modelling the whole image, as a set of layers. The representation includes a single background layer and one layer for each object. Each layer consists of a shape prior, motion model, and layer appearance. A layering is performed by first compensating the background motion modelled by projective motion, such that the object's motion can be estimated from the compensated image using 2D parametric motion. Then, each pixel's probability of belonging to a layer is computed based on the object's previous motion and shape characteristics. Any pixel far from a layer is assigned a uniform background probability. Later, the object's appearance (intensity, color) probability is coupled with to obtain the final layer estimate. Due to the difficulty in simultaneously estimating the parameters, the authors individually estimate one set while fixing the others. For instance, they first estimate layer ownership using intensity for each pixel, then they estimate the motion using appearance probabilities and finally update layer ownership using this motion. The unknowns for each object are iteratively estimated until the layer ownership probabilities are maximized.

Isard and MacCormick [71] propose joint modelling of the background and foreground regions for tracking. The background appearance is represented by a mixture of Gaussians for the background over small patches. The appearance of all foreground objects is modelled by a mixture of Gaussians. The shape of objects is modelled as cylinders. They assume the ground plane is known, thus the 3D object positions can be computed. Tracking is achieved by using particle filters, where the state vector includes the 3D position, shape and velocity of all objects in the scene. They propose a modified prediction and correction scheme for particle filtering, which can increase or decrease the size of the state vector to include or remove objects. The method can also tolerate occlusion between objects. However, the maximum number of objects in the scene is required to be predefined. Another limitation of the approach is the use of same appearance model for all foreground objects and it requires training to model the foreground regions.

Tracking by Multi-view Appearance Models

In the aforementioned tracking methods, the appearance models, like histograms and templates are usually generated online. Thus these models represent the information gathered about the object from the most recent observations. The objects may appear different from different views and if the object view changes dramatically during tracking, the appearance model may no longer be valid and the track of the object might be lost. To overcome this problem, different views of the object can be learned off-line and used for tracking.

Black et. al [14] propose a subspace based approach to compute the affine transformation from the current image of the object to the image reconstructed using eigenvectors. First, a subspace representation of the appearance of an object is built using Principal Component Analysis, then the transformation from the image to the Eigenspace is computed by minimizing the so-called subspace constancy equation, which evaluates the difference between the image reconstructed using the eigenvectors and the input image. Minimization is performed in two steps: finding subspace coefficients and computing affine parameters. In the first step, the affine parameters are fixed and the subspace coefficients are computed. In the second step, using the new subspace coefficients, affine parameters are computed. Based on this, tracking is performed by estimating the affine parameters iteratively until the difference between the input image and the projected image is minimized.

Avidan [13] uses a SVM classifier for tracking. SVM is a general classification scheme that, given a set of positive and negative training examples, finds the best separating hyper-plane between the two classes [69]. During testing, the SVM gives a score to the test data indicating the degree of membership of the test data to the positive class. For SVM based trackers, the positive examples consist of the images of the object to be tracked and the negative examples of all other things that are not to be tracked. Generally negative examples consist of background regions that could be confused with the object. Avidan's tracking method, instead of trying to minimize the intensity difference of a template from image regions, tries to maximize the SVM classification score over image regions in order to determine the location of the object.

Discussion and Evaluation

Geometric shape models are suitable representations for rigid objects, though they have also been used in the context of tracking non rigid objects. These tracking methods model the appearance by templates, probability densities or multi-view models. The motion is defined in terms of translation, translation+rotation, affine or projective transformation.

The methods that use gradient descent based minimization for motion estimation require overlap between object regions in successive frames, however this assumption is valid for many tracking scenarios.

Table 1.4 qualitatively compare tracking algorithms in the section according to the criteria: tracking single or multiple objects, ability to handle occlusion, requirement for training, and requirement of manual initialization.

Table 1.4: Qualitative Comparison of Geometric Model Tracking

	Obj No.	Train	Occlusion	User Init.
Simple template matching	single	No	Partial	Yes
Mean shift [11]	single	No	Partial	Yes
KLT [42]	Single	No	Partial	Yes
Appearance Tracking [66]	Single	No	Partial	Yes
Layering [67]	Multi	No	Full	No
Bramble [68]	Multi	Yes	Full	No
EigenTracker [14]	Single	Yes	Partial	Yes
SVM [13]	Single	Yes	Partial	Yes

1.3.3 Tracking by Contour Evolution

The goal of a contour based object tracker is to find the boundary between the object and the background in each frame, such that the object region is tightly enclosed within the contour. We can categorize the contour based object trackers into two categories based on how the contour is evolved. The first category of contour trackers uses state space models to evolve the contour. The second category performs direct minimization of a contour energy functional.

Tracking using State Space Models

In this section, we will discuss tracking object contours that use the probabilistic state space approaches to track the object. The object's state is defined in terms of the shape and the motion parameters of the contour. The state is updated at each time instant such that the contour's a posteriori probability is maximized. The posterior probability depends on the prior state and the current likelihood, which is usually defined in terms of the distance of the contour from observed edges.

Terzopoulos et. al [70] define the object state by the dynamics of the control points. The dynamics of the control points are modelled in terms of a spring model, which moves the control points based on the spring stiffness parameters. The new state (spring parameters) of the contour is predicted using Kalman filters. The correction step uses the image observations which are defined in terms of the image gradients.

Isard et. al [71] define the object state in terms of spline shape parameters and affine motion parameters. The measurements consist of image edges computed in the normal direction to the contour. The state is updated using a particle filter. In order to obtain initial samples for the filter, they compute the state variables from the contours extracted in consecutive frames during a training phase. During the testing phase, the current state variables are estimated through particle filtering based on the edge observations along normal lines at the control points on the contour.

MacCormick and Blake [72] extend the particle filter based object tracker in [71] to track multiple objects by including the ”exclusion principle” for handling occlusion. The exclusion principle integrates into the sampling step of the particle filtering framework, such that for two objects, if a feature is lying in the observation space of both objects then it contributes more to the samples of the object which is occluding the other object. Since the exclusion principle is only defined between two objects, this approach can track at most two objects undergoing occlusion at any time instant.

Chen et al. [73] propose a contour tracker, where the contour is parameterized as an ellipse. Each contour node has an associated Hidden Markov Model and the state of each Hidden Markov Model is defined by the points lying on the lines normal to the contour control point. The observation likelihood of the contour depends on the background and the foreground partitions defined by the edge along the normal line on contour control points. The state transition probabilities of the Hidden Markov Model are estimated using the Joint Probability Data Association Filtering. Given the observation likelihood and the state transition probabilities the current contour state is estimated using the Viterbi algorithm [74]. After the contour is approximated, an ellipse is fit to enforce the elliptical shape constraint.

The methods discussed above represent the contours using explicit representation, e.g., parametric spline. Explicit representations do not allow topology changes, such as region split or merge [75]. Next, we will discuss contour tracking methods based on direct minimization of an energy functional. These methods can use implicit representations and allow topology changes.

Tracking by Direct Minimization of Contour Energy Functional

In this category, algorithms evolve the contour onto the object region by minimizing the energy functional. The contour energy is computed using temporal information in the form of either the temporal gradient [79, 18], or appearance models computed from object and background regions [36].

Contour tracking using temporal image gradients is motivated by the extensive work on computing the optical flow: $I^{t+1}(x, y) - I^{t}(x - u, y - v) = 0$, where I is the image, t is the time, and (u, v) is the flow vector in the x and the y directions. Bertalmio [76] use this constraint to evolve the contour in consecutive frames. Their objective was to compute u and v iteratively for each contour position using the level set representation. At each iteration, contour speed in the normal direction $\vec{n}$ is computed by projecting the gradient magnitude on $\vec{n}$. The authors use two energy functionals, one for contour tracking, E_t and one for intensity morphing, $E_m : E_m(\Gamma) = \int_0^1 E_{im}(v)ds$ and $E_t(\Gamma) = \int_0^1 E_{ext}(v)ds$. The intensity morphing functional, which minimizes intensity changes in the current and the previous frames, $\bigtriangledown I_t = I_t - I_{t-1}$ on the hypothesized object contour:

$$\frac{\partial F(x, y)}{\partial t} = \bigtriangledown I_t(x, y) \| \bigtriangledown F(x, y) \|$$

is coupled with the contour tracking equation:

$$\frac{\partial \phi(x, y)}{\partial t} = \bigtriangledown I_t(x, y) \vec{n}_F \vec{n}_\phi \| \bigtriangledown F(x, y) \|$$

and both functionals are minimized simultaneously. For instance, if $\bigtriangledown I_t(x, y) \geq 0$, then the contour moves with the maximum speed in its normal direction and $\bigtriangledown I_{t-1}(x, y)$ is morphed into $\bigtriangledown I_t(x, y)$. On the other hand, if $\bigtriangledown I_t(x, y)$ is negligible, then the evolution speed will be zero.

Similarly, Mansouri [77] uses the optical flow constraint for contour tracking. In contrast to [77] which computes the flow only on the object boundary, his approach is motivated by computing the flow vector for each pixel inside the complete object region in a circular neighborhood with radius r using a brute-force-search. Once the flow vectors are computed, the contour energy, which is based on the brightness constancy constraint, is evaluated. This process is iteratively performed until the energy is minimized.

In [18], Cremers and Schnorr use the optical flow as an feature for contour evolution, such that an object can only have homogeneous flow vectors inside the region. Their energy is a modified form of the common Mumford-Shah energy [78], which evolves the contour until a region with homogeneous flow vectors is encountered. They also incorporated the shape priors to better estimate the object shape. The shape priors are generated from a set of object contours, such that each control point on the contour has an associated Gaussian with a mean of the spatial positions of the corresponding control points on all the contours along with a standard deviation.

Tracking methods using region priors do not explicitly use the temporal information in the form of the brightness constancy constraint. In contrast, they use appearance priors generated online. Tracking is performed by initializing the contour in the current frame with its previous position. Ronfrad [36] propose a region based contour tracking method. His functional is defined based on the piecewise stationary image models formulated as Ward distances, which is a measure of image contrast. Since the resulting energy does not have an analytical form, each contour point is evolved individually based on its local neighborhood.

Discussion and Evaluation

The most important advantage of a contour tracker is that it can model a large variety of object shapes. Contours are represented by explicit (control points and splines) or implicit (level sets) representations. The use of these representations depends on the context of the application. For instance, in most cases for tracked objects that do not split or merge, explicit representation is usually suitable for tracking. However, in some applications such as surveillance, it is important to keep track of a person leaving an object behind. In the context of contour tracking, when a person leaves an object, part of the object contour will be placed on the left object (region split). Topology changes like region split or merge can be handled well by implicit representations.

Contour trackers are employed when tracking the complete region of an object is required. In the context of region tracking, the precision and recall measures are defined in terms of the intersection of the hypothesized and correct object regions. The precision is the ratio of the intersection to the hypothesized region and recall is the ratio of the intersection to the ground truth. Qualitatively, the contour based methods can be compared on the basis of requirement of training and occlusion handling. Moreover some algorithms only use information on the contour boundary for evolution while other use the complete region. Generally the region based approaches are more resilient to noise. A qualitative comparison of contour based approaches is given in Table 1.5.

1.4 Conclusion

In this chapter, we present an extensive survey of object tracking methods. We divide the tracking methods into three categories based on the use of object representations: namely, methods establishing point correspondence, methods using geometric models and methods using contour evolution. Note that all these classes require object detection at some point. For instance, the point trackers require detection in every frame, whereas geometric region or contours based trackers require detection only when the object first appears in the scene. Recognizing the importance of

Table 1.5: Qualitative Comparison of Contour Based Tracking

	Obj No.	Train	Occlusion	Region/Boundary
Ref[73]	Single	Yes	None	Boundary
Ref[71]	Single	Yes	None	Boundary
Ref[72]	Multi	Yes	Full	Boundary
Ref[82]	Multi	Yes	None	Boundary
Ref[77]	Single	No	None	Region
Ref[76]	Single	No	None	Region
Ref[37]	Single	No	None	Boundary
Ref[83]	Single	Yes	Partial	Region

object detection for tracking systems, we include a short discussion on popular object detection methods. We provide detailed summaries of object trackers, including discussion on the object representations, motion models, and the parameter estimation schemes employed by the tracking algorithms. Moreover, we describe the context of use, degree of applicability, evaluation criteria and qualitative comparisons of the tracking algorithms.

Chapter 2

Implementation and Performance of Classic Tracking Algorithms

The object tracking techniques introduced in previous sections have been intensively studied. There are many variants exist in each category of the tracking approaches. Furthermore, there are more papers and more variants every day. But the core ideas behind every variant remain unchanged, hence cutting to the chase right up front is the best way, in our opinions, to expose those innovative ideas to the audiences. So in the following subsections, we will discuss those central ideas, their implementations and performances in a fairly straightforward manner.

2.1 Tracking by Background Modelling and Subtraction

The Basic approach is to maintain a model of the static background and compare the current frame with the background to locate moving foreground objects (Figure 2.1). Background Subtraction is probably the simplest idea for tracking moving objects, yet it is one of the most effective and robust algorithms exists.

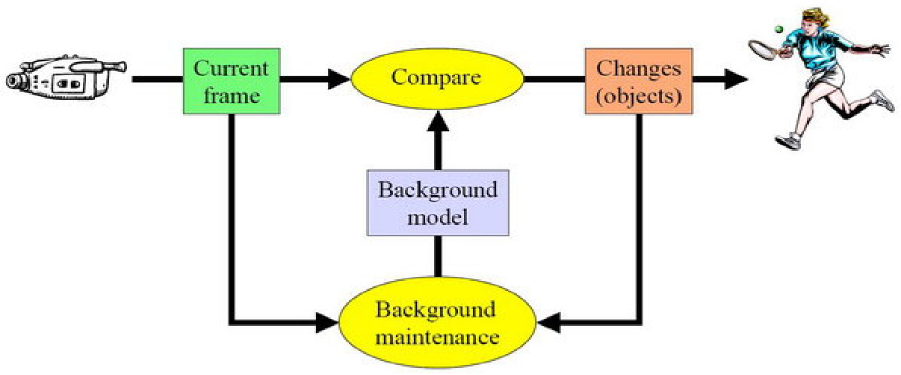


Figure 2.1: The overview of background subtraction.

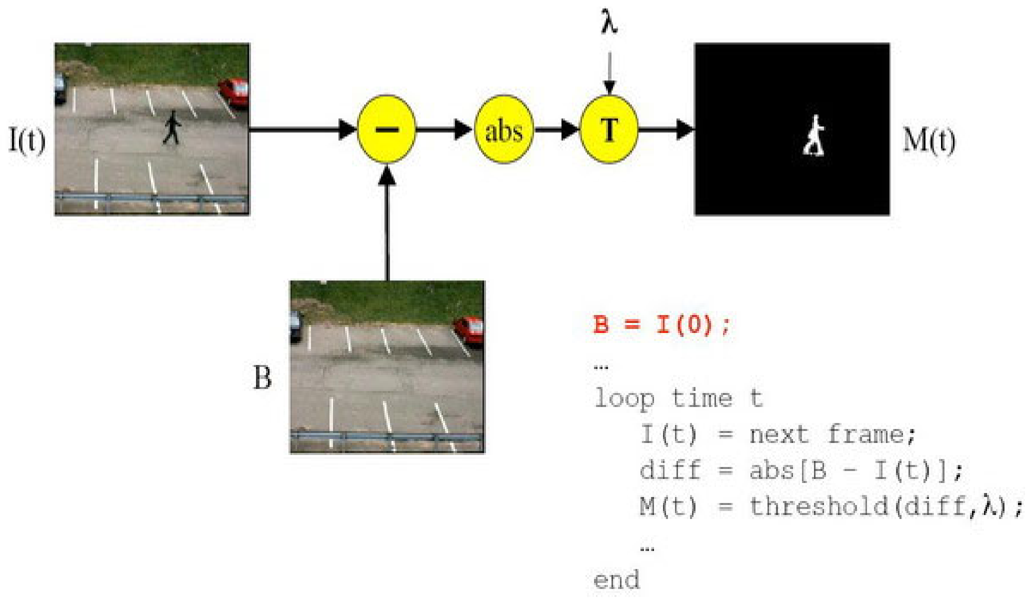
The static background subtraction method assumes that the background image remains static throughout the entire video sequence. The background is computed by taking the average of several frames in the video sequence in which no objects present. Pixels are labelled as object (1) or not object (0) based on thresholding the absolute intensity difference between the current frame and background (Figure 2.1).

We implemented a simple background subtraction 2D people tracker. It bases on background subtraction, combining gap-spanning connected component and size filter techniques. This 2D tracker is capable of tracking multiple objects (e.g., people, vehicles) moving within the field of view (FOV). It assumes that the background should be a constant, the camera should be fixed and the movement of the objects should be salient.

The core algorithm contains four steps:

Step 1: Simple Background Subtraction. Suppose we have several frames that contain only the background. By taking the average of these frames, we can obtain an approximation of the background. Then the moving foreground objects can be identified by comparing the current frame with the approximate background Figure 2.2. Pixels are labelled as object(1) or not object(0) based on thresholding the absolute intensity between the current frame and background. Figure 2.3 shows the background subtraction results.

Step 2: Gap-spanning connected component. Noise pixels in the difference image are first



```
B = I(0);
...
loop time t
   I(t) = next frame;
   diff = abs[B - I(t)];
   M(t) = threshold(diff,λ);
   ...
end
```

Figure 2.2: Simple background subtraction.

removed by median filter and erosion. Then, a gap-spanning connected component algorithm is used to find blobs (there may be more than one).

Due to the small motion in the background and accidental similarity between moving objects and background, we obtain only a non-perfect detection of moving objects from step 1, i.e. some true detections are eliminated and the detected foreground objects are split. We can fill those gaps by using morphological operations (e.g., dilate). After enhancing the detection, the connected components are labelled by scanning the difference image, pixel-by-pixel (from top to bottom and left to right) to identify connected pixel regions, i.e. regions of adjacent pixels which share the same set of intensity values V. (For a binary image V=1; however, in a gray level image V will take on a range of values, for example: V=51, 52, 53, ..., 77, 78, 79, 80.). Figure 2.4 illustrates the gap-spanning connected component algorithm.

Step 3: Size filter. Following step 2, we measure a set of properties for each labelled connected components (blobs). The good properties for tracking are area, color, velocity, etc. We find that the ”area” property alone will suffice for most cases. By selecting only the blobs whose areas

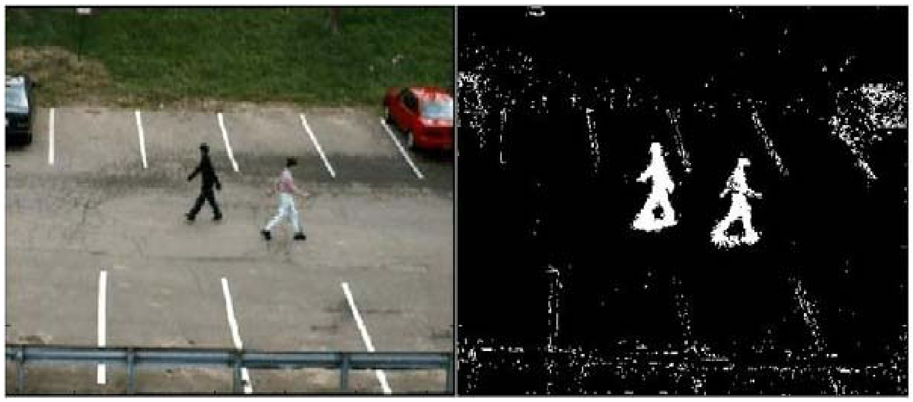


Figure 2.3: Background subtraction results (a) Frame No.211, (b) The difference image between (a) and background.

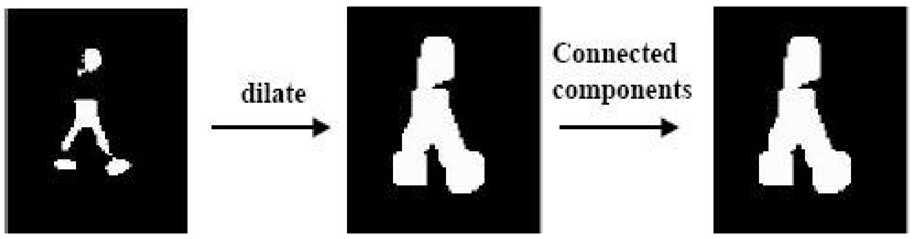


Figure 2.4: Gap spanning connected components.

are larger than a threshold (450 was used in our implementation.), we size-filtered the difference image. The blobs that survive this step are considered as saliently moving objects (Figure 2.5).

Step 4. Compute the size of each blob and plot the bounding boxes around the moving objects.

Simple Background Subtraction is a reasonable solution to extracting the shape of moving objects, as we can see in Figure 2.5. One important drawback of simple background subtraction is that it is very sensitive to changing illumination and unimportant movement of the background, for example, trees blowing in the wind, reflections of sunlight off of cars or water. Another serious drawback is that it cannot handle movement of the camera. These are the reasons why statistical background subtraction is proposed. As we discuss in the previous section, there are two models for representing the background: one is parametric and the other is non-parametric. The parametric

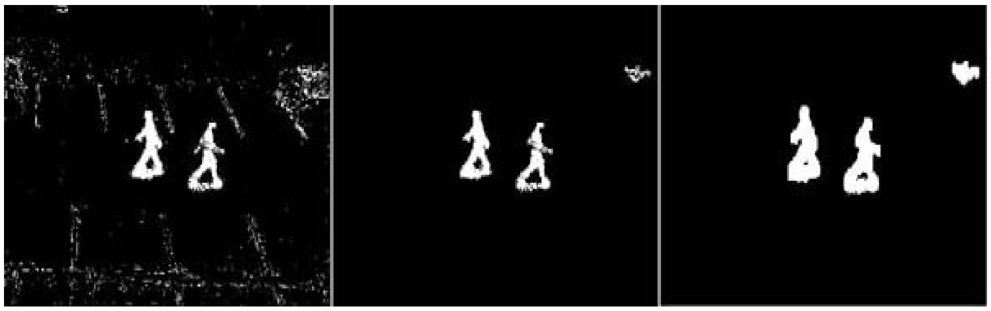


Figure 2.5: Grouping pixels into blobs. (a) The difference image. (b) after median filter, erosion and size filter. (c) dilate and label the connected components.

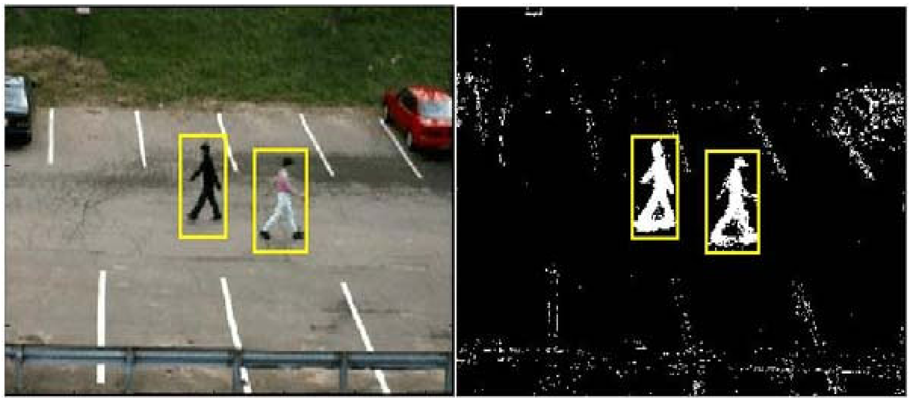


Figure 2.6: Frame No.211

technique models the pixel intensity by a mixture of K gaussian distributions (K is a small number from 3 to 5) to model variations in the background:

$$Pr(x_t) = \sum_{j=1}^{K} \frac{w_j}{(2\pi)^{\frac{d}{2}} |\Sigma_j^{-1}|} e^{-\frac{1}{2}(x_t-\mu_j)^T \Sigma_j^{-1}(x_t-\mu_j)} \tag{2.1}$$

where w_j is the weight, μ_j is the mean and $\Sigma_j = \sigma_j^2 I$ is the covariance for the jth distribution. The K distributions are ordered based on $\frac{w_j}{\sigma_j^2}$ and the first B distributions are used as a model of the background of the scene, where B is estimated as

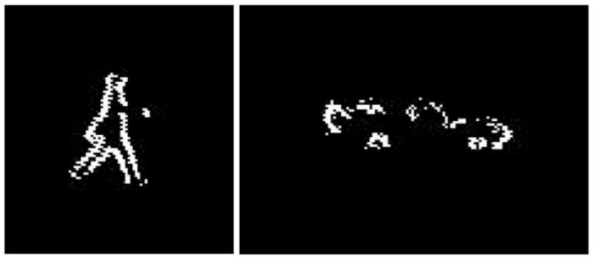


Figure 2.7: Only pixels near the boundary are detected on the object

$$B = argmin_{\epsilon}\left(\frac{\sum_{j=1}^{b} w_j}{\sum_{j=1}^{K} w_j} > T\right) \tag{2.2}$$

The threshold T is the fraction of the total weight given to the background model. Background subtraction is performed by marking any pixel that is more than 2.5σ away from any of the B distributions as a foreground pixel.

In the case where the background has very high frequency variations, this parametric method fails to achieve sensitive detection. When all variations occur in a very short period of time (e.g., 30 seconds), modelling the background variations with a small number of gaussian distributions will not be accurate. Furthermore, the very wide background distribution will result in poor detection because most of the gray level spectrum would be covered by the background model.

Non-parametric model, on the other hand, can accurately model the background and adapt very quickly to background changes. Let $x_1, x_2, ..., x_N$ be a recent sample of intensity values for a pixel. Using this sample, the probability density function that this pixel will have intensity value x_t at time t can be non-parametrically estimated using the kernel estimator K as:

$$Pr(x_t) = \frac{1}{N}\sum_{i=1}^{N} K(x_t - x_i) \tag{2.3}$$

Usually, the kernel estimator function, K, is chosen to be a gaussian density function $N(0, \Sigma)$. We can also reasonably assume independence between the different color channels with a different kernel bandwidths σ_j^2 for the jth color channel.

Using this probability estimate, the pixel is considered a foreground pixel if $Pr(x_t) < T$ where T is a global threshold over all the image that can be adjusted to achieve a desired percentage of false positives. It is easy to see that nonparametric model with a gaussian kernel estimator function is actually a generalization of the Gaussian mixture model, where each single sample of the N samples is considered to be a Gaussian distribution $N(0, \Sigma)$ by itself. This allows us to estimate the density function more accurately and depending only on recent information from the sequence. This also enables the model to quickly forget about the past and concentrate more on recent observation, which typically require large amounts of data to be both accurate and unbiased.

An extension of simple background subtraction to mobile camera can be obtained by finding the absolute difference between the current frame and the previous frame instead of the background frame (Frame Differencing).

Frame differencing is very quick to adapt to changes in lighting or motion. Objects that stop are no longer detected. Objects that start up do not leave behind ghosts. However, frame differencing only detects the leading and trailing edge of a uniformly colored object. As a result (Figure 2.7), very few pixels on the object are labeled, and it is very hard to detect the entire object that is moving before the camera.

One way to solve this problem is to adjust the temporal scale (frame rate) at which we perform two-frame differencing, define:

$$D(N) = |I(t) - I(t + N)| \tag{2.4}$$

Figure 2.8 shows the results of frame differencing with $N = -1, -3, -5, -9, -15$. Obviously, we can capture more complete object silhouette as N increases. But we also see two copies of the same object (one where the object used to be and one where it is now), this observation motivates the three-frame differencing.

In three-frame differencing, we first choose a good frame-rate N which depends on the size and speed of the object of interest. Then, $D(+N)$ and $D(-N)$ are computed. $D(+N)$ contains the object's current position and its future position, $D(-N)$ contains the object's current position

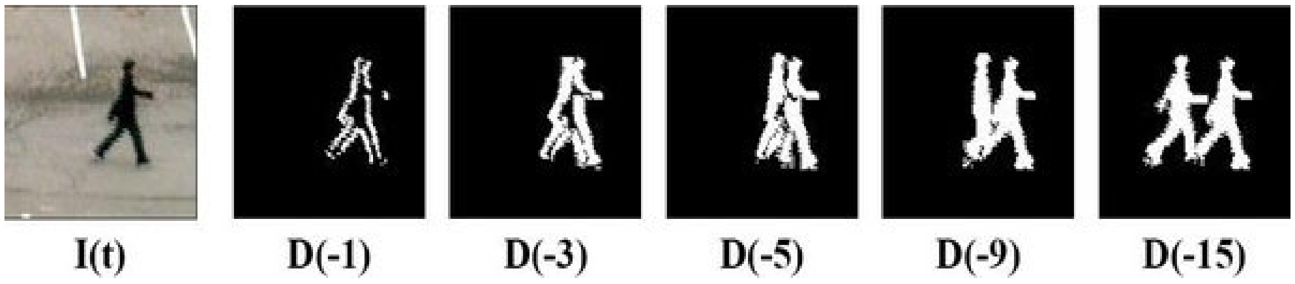


Figure 2.8: Frame Differencing with N = -1, -3, -5, -9, -15

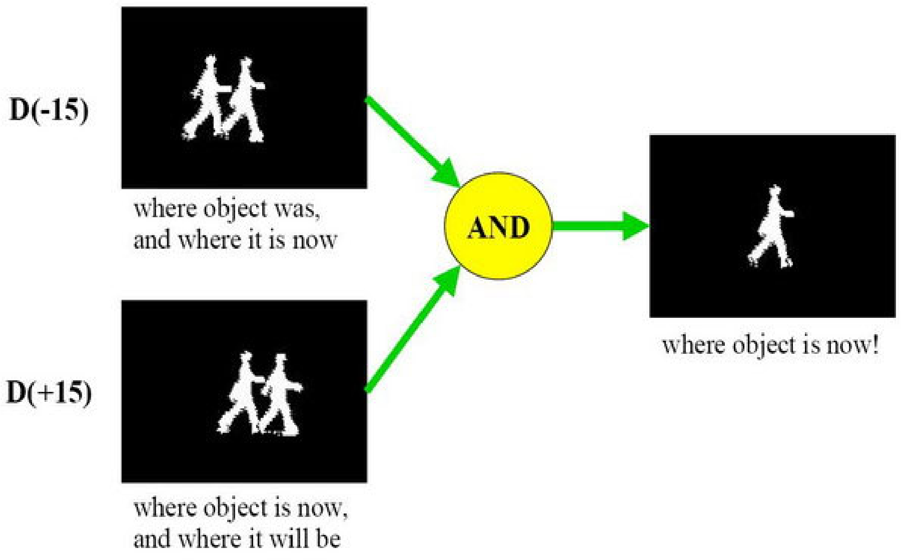

D(-15)
where object was, and where it is now
D(+15)
where object is now, and where it will be
AND
where object is now!

Figure 2.9: The AND operation

and its past position. Next, the logical AND operation is taken between $D(+N)$ and $D(-N)$. The AND operation results in an image with only the object in its current position (Figure 2.9). The following steps after this is basically the same as those in simple background subtraction.

We implement three-frame differencing algorithm to track an aerial video sequence from naval data test set (Figure 2.10). The moving vehicles present throughout the entire sequence, so there is no way to capture the background. But three-frame differencing works well here. $N = 3$ is chosen as a good frame rate.

The trackers we implement in this section is perhaps the simplest tracking implementation and

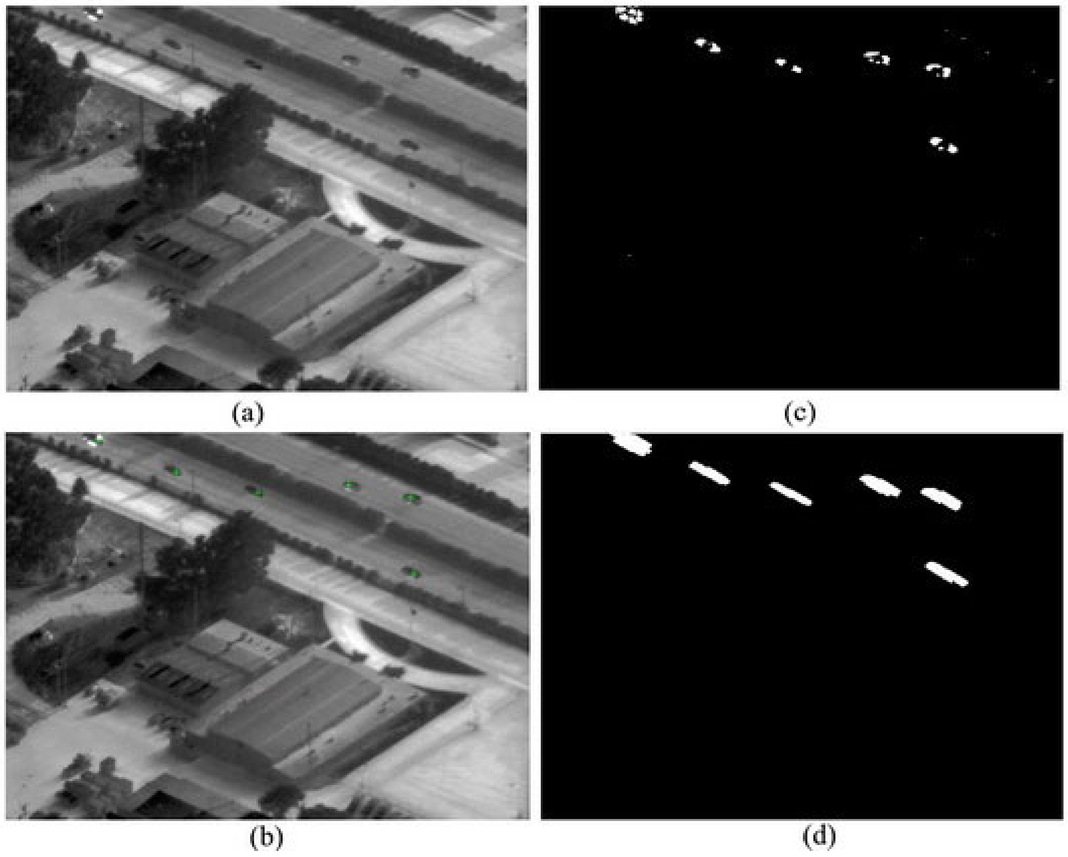


Figure 2.10: Frame Differencing Tracking with $N = 3$. (a) input image; (b)output image; (c) frame differencing result; (d) gap-spanning (c)

in many application scenarios, especially those with camera fixed, the core algorithm provides the first step for tracking and tracking initializations.

2.2 Tracking by Point Correspondence

In Point Correspondence tracking, Objects detected in consecutive frames are associated based on the previous object state, which can include motion and shape characteristics of the object. This approach requires an external mechanism to detect the objects in every frame. This section

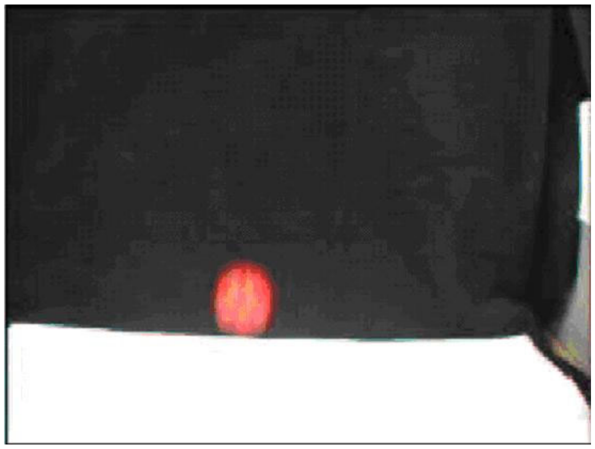


Figure 2.11: A snapshot of the video sequence of a bouncing ball

introduces the implementation of point correspondence tracking through a very simple example: tracking a bouncing ball. We will stress on the statistical method. Figure 2.11 is a snapshot of this video sequence.

We use three different tracking techniques which have been discussed in the previous section:

- *Background Subtraction Tracking.* Removal of irrelevant background and detection of the ball. It can also be utilized to detect the objects and capture the object's state in every frame.
- *Kalman Filter Tracking.* Tracking noisy motion.
- *Particle Filter Tracking.* Coping with events and noise with condensation tracking

2.2.1 Background Subtraction Tracking

Following the discussion before, this implementation is easy enough to understand and is the standard initializing procedure for many more sophisticated tracking algorithms:

Step 1: Compute the background image by averaging the first few frames contain only the static background.

Step 2: For each frame, we subtract background from it and obtain the difference image, then erode the difference image to remove small noise, finally select the largest, valid object from this difference image (By valid, we mean the area of the object is larger than a certain threshold, e.g., half of the area of the ball.)

Step 3: Compute the center of mass and radius of this largest, valid object and plot the contour of the ball.

Background subtraction method is very sensitive to noise, random perturbation, motion blur and poor contrast. One of the ways to improve the result is to incorporate motion model into the tracking process.

2.2.2 Kalman Filter Tracking

Kalman filter tracking treat the tracking problem as inferring the object state by taking the measurement and the model uncertainties into account. We've already discussed the technique, so now we go directly into implementation. Most important thing is to put the physical model of the ball moving under gravity into the Kalman filter:

PHYSICAL MODEL OF THE FALLING BALL:

Ball Position: $\vec{p}_t = (x_t, y_t)$ — Measured Data

Position update: $\vec{p}_t = \vec{p}_{t-1} + \vec{v}_{t-1} \triangle t$

Velocity update: $\vec{v}_t = \vec{v}_{t-1} + \vec{a}_{t-1} \triangle t$,$\vec{a}_{t-1} = (0, g)^T$

State Vector: $\vec{x}_t = (x_t, y_t, v_{x,t}, v_{y,t})$

INITIALIZATION:

The initial state vector can be chosen randomly. Based on the physical model of the falling ball (Newton's law), we can decide the rest of the parameters that are necessary for initializing Kalman filter:

$$A = \begin{pmatrix} 1 & 0 & \triangle t & 0 \\ 0 & 1 & 0 & \triangle t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$B\vec{u_t} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ g \triangle t \end{pmatrix}$$

We use $\triangle t$=1 in the program.

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$Q = \begin{pmatrix} 0.01 & 0.01 \\ 0.01 & 0.01 \end{pmatrix}$$

$$R = \begin{pmatrix} 0.01 & 0.01 \\ 0.01 & 0.01 \end{pmatrix}$$

$$P = \begin{pmatrix} 100 & 100 \\ 100 & 100 \end{pmatrix}$$

PREDICTION:

1. Predict likely state given what we already know:

$$\vec{y_t} = A\vec{x}_{t-1} + B\vec{u}_{t-1}$$

2. Predict error of the predicted state:

$$E_t = AP_{t-1}A^T + Q \qquad (2.5)$$

3. Estimate correction gain:

$$K_t = E_t H^T H E_t H^T + R^{-1} \tag{2.6}$$

4. Estimate new state given prediction and correction from observations:

$$\vec{x}_t = \vec{y}_t + K_t(\vec{z}_t - H\vec{y}_t) \tag{2.7}$$

5. Estimate error of estimated state:

$$P_t = (I - K_t H)E_t \tag{2.8}$$

Figure 2.12(a,b) shows the tracking results of the Kalman filter, in which the ball is successfully tracked. We also show some failures in Figure 2.12(c,d). The failure arises mainly because of the two situations: 1. when the ball stops falling and start to bounce back. In this situation, Kalman filter processing model assumes the ball is always falling. So the model is predicting the movement to one direction, while the observation is seeing it moving in the other direction. Kalman filter combines those two pieces of information, but the result is that the estimated state vector has been a little bit lower than it really ought to be. 2. When the ball has stopped bouncing and sat on the table, the model is still predict the ball to be falling, the situation is very similarly to the first one. That's why the particle filter tracking comes in. The particle filter tracking, a.k.a. condensation tracking, uses several key ideas to overcome some of the problems we had with the Kalman filter.

2.2.3 Particle Filter Tracking

One of the key ideas of the particle filter tracking is to keep multiple hypotheses. It is very convenient in this particular case: as well as the ball falling at any particular time, at the next time, the ball could bounce or stop, so we can use multiple hypotheses to keep tracking of those different states. Noise observation also needs us to keep multiple hypotheses around and choose the one that best fit the current time frame.

The outline of particle filter theory is:

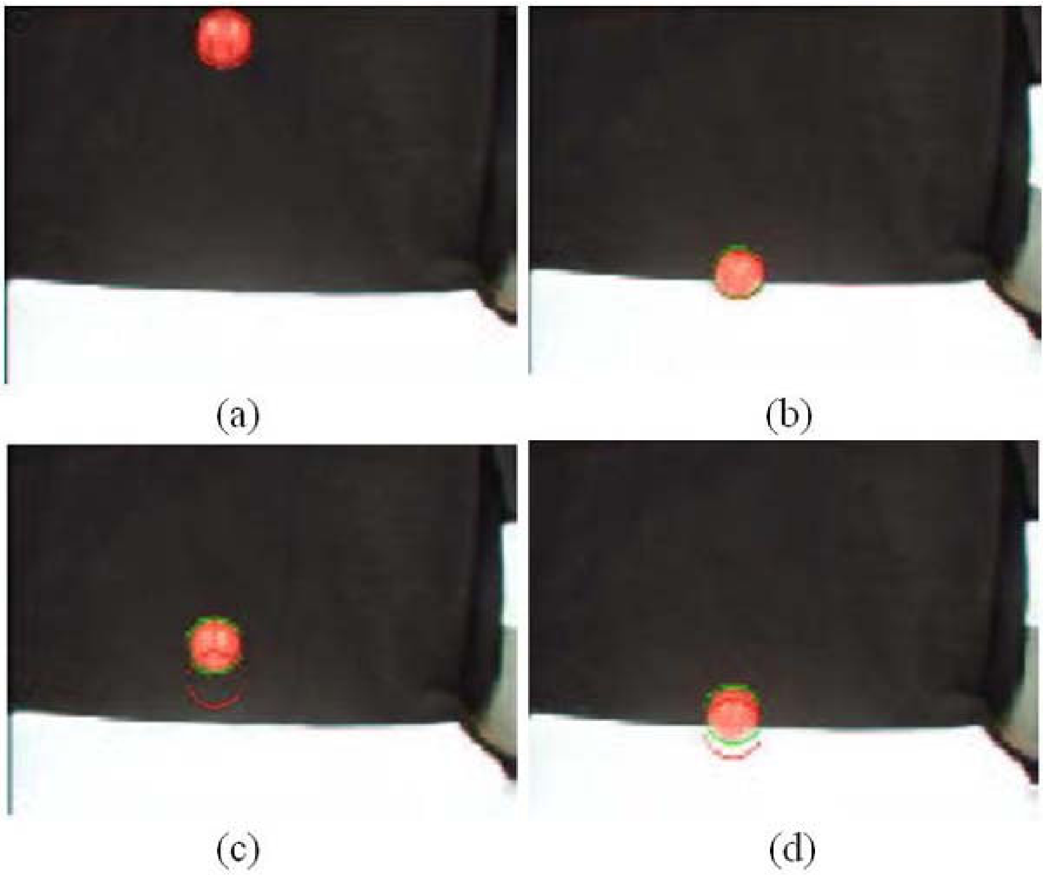


Figure 2.12: (a,b) Kalman filter success, (c,d)Kalman filter failure.

Given a set of N hypotheses at time $t-1$, $H_{t-1} = \{\vec{x}_{1,t-1}, \vec{x}_{2,t-1}, ..., \vec{x}_{N,t-1}\}$ with associated probabilities $\{p(\vec{x}_{1,t-1}), p(\vec{x}_{2,t-1}), ..., p(\vec{x}_{N,t-1})\}$, we repeat N times to generate a new set of hypotheses H_t:

1. Randomly select a hypothesis $\vec{x}_{k,t-1}$ from H_t with probability $p(\vec{x}_{k,t-1})$

2. Generate a new state vector $\vec{s}_{t-1}$ from H_t with a distribution centered at $\vec{x}_{k,t-1}$.

3. Get new state vector using dynamic model $\vec{x}_t = f(\vec{s}_{t-1})$ and Kalman filter.

4. Evaluate probability $p(\vec{z}_t|\vec{x}_t)$ of observed data given state $\vec{x}_t$

Next, we will discuss how particle filter could improve the tracking of a bouncing ball.

1. In our implementation, we maintain a set of 100 hypotheses of ball motion vector, at each time we select one of them by the probability of the vector.

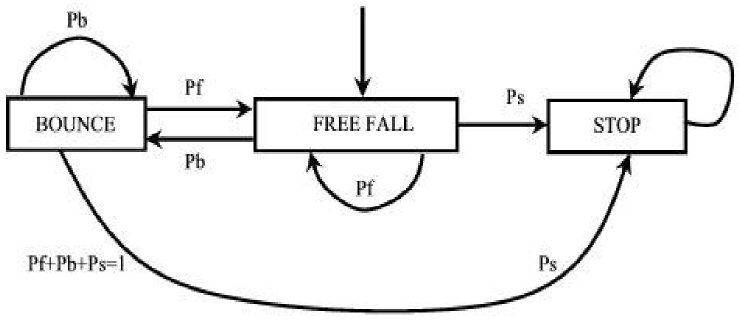


Figure 2.13: model transition diagram for the falling ball

2. we use estimated covariance matrix P to create state samples $\vec{s}_{t-1}$

3. we decide, given the state sample $\vec{s}_{t-1}$, which physical situation the ball should be in (Bouncing, falling or stop). Each physical situation has a probability associated with it (Figure 2.13). In the program, we set Pb = 0.3, Ps = 0.05.

If it is in STOP situation: set the vertical speed to zero,

If it is in BOUNCE situation: set $v_y = -0.7 \cdot v_y$,

Then use Kalman filter,

4. we estimate hypotheses goodness by $\frac{1}{\|H\vec{x}_t - \vec{z}_t\|^2}$ and normalize it to estimate hypotheses probability.

5. Select top-weighted hypotheses and draw the estimated contour of the ball.

Figure 2.14 shows that Kalman filter failures are fixed by particle filter tracking.

2.3 Tracking by Density Based Appearance Models

In this section, we implemented an efficient, modular target tracking algorithm proposed by Comaniciu and Meer [11].

The target objects are represented by regularized histogram, i.e., target model, computed from a target region (e.g., a person, a football or a plane) which should be specified by the user at the beginning. Then, the algorithm will search for an optimal target candidate in the next frame, which

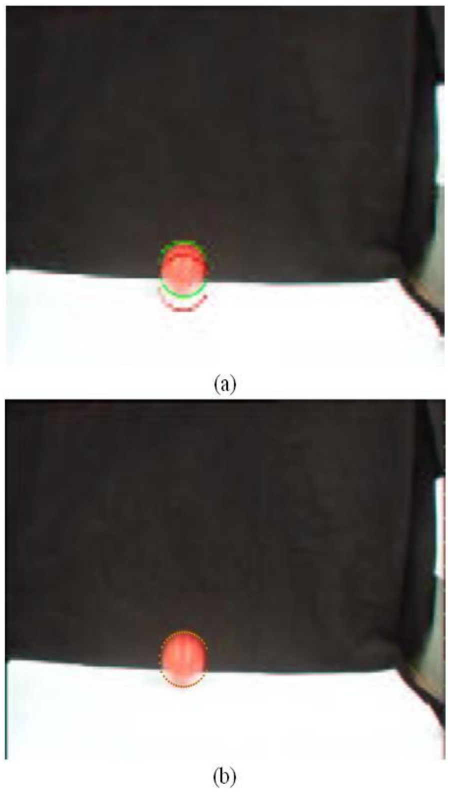


Figure 2.14: (a) Kalman filter failed in this frame, (b) Particle filter succeeded in the same frame

has the largest similarity with target model . The similarity is measured by Bhattacharyya coefficient. Experimental results show that the method has successfully coped with complex camera motion, partial occlusion of the target, presence of significant clutter, and large variations in target scale and appearance.

The target model is represented by its probability density function (pdf), In our implementation, we use the color pdf of the target.

The target model is:

$$\hat{\mathbf{q}} = \hat{q_u}, u = 1...m \tag{2.9}$$

$$\sum_{u=1}^{m} \hat{q_u} = 1 \tag{2.10}$$

where m is the number of bins in the histogram.

From pattern recognition, we know that $\hat{q_u}$ can be estimated using the following formula:

$$\hat{q_u} = C \sum_{i=1}^{n} k(\|y - x_i\|^2)\delta[b(x_i) - u] \tag{2.11}$$

where n is the number of the pixels in the region, y is the center of the region, δ is the Kronecker delta function, C is the normalization constant.

The target candidate is represented by:

$$\hat{\mathbf{p}}(y) = \hat{p_u}(y), u = 1...m \tag{2.12}$$

$$\sum_{u=1}^{m} \hat{p_u} = 1 \tag{2.13}$$

where m is the number of bins in the histogram, y is the center of the region.

$\hat{p}_u(y)$ can also be estimated by a formula very similar to $\hat{q_u}$'s:

$$\hat{p_u}(y) = C_h \sum_{i=1}^{n_h} k(\|\frac{y - x_i}{h}\|^2)\delta[b(x_i) - u] \tag{2.14}$$

where C_h is the normalization constant, n_h is the number of pixels in an image.

The similarity between $\hat{p_u}(y)$ and $\hat{q}_u$ is measured by Bhattacharyya coefficient and the center y of the target candidate in the next frame is localized by maximizing the Bhattacharyya coefficient.

By employing the mean shift procedure, the optimal new location for the center $\hat{y}_1$ can be obtained by the following formula:

$$\hat{y}_1 = \frac{\sum_{i=1}^{n_h} x_i w_i g(\|\frac{\hat{y}_0 - x_i}{h}\|^2)}{\sum_{i=1}^{n_h} w_i g(\|\frac{\hat{y}_0 - x_i}{h}\|^2)} \tag{2.15}$$

$$w_i = \sum_{u=1}^{m} \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{y}_0)}}\delta[b(x_i) - u] \tag{2.16}$$

where $\hat{y}_0$ is the previous location of the center, $g(x) = -k'(x)$.

The above formula will be further reduced to a much simpler form if we choose Epanechnikov kernel, hence (2.15) becomes

$$\hat{y}_1 = \frac{\sum_{i=1}^{n_h} x_i w_i}{\sum_{i=1}^{n_h} w_i} \tag{2.17}$$

The complete algorithm can be implemented as follows:

Given the target model $\hat{\mathbf{q}} = \hat{q_u}, u = 1...m$ and its location $\hat{y}_0$ in the previous frame.

1. Initialize the location of the target in the current frame with $\hat{y}_0$, compute:

$$\hat{\mathbf{p}}(y_0) = \hat{p_u}(y_0), u = 1...m \tag{2.18}$$

2. Derive the weights $w_{i=1...n_h}$.

3. Find the estimated location of the target in the next frame by:

$$\hat{y}_1 = \frac{\sum_{i=1}^{n_h} x_i w_i}{\sum_{i=1}^{n_h} w_i} \tag{2.19}$$

We tested this algorithm on aerial video sequences from Naval Data Test Set (Figure 2.15-2.20). In those video sequences, the camera moves fast and complexly, hence the background is very cluttered. Also, in the thermal video sequences, due to the huge similarity between the background and the target, the moving target is often ”camouflaged” or occluded for a while by the background. Experimental results show that, as long as the camouflaging period is not too long, the algorithm can still keep track of the moving target after occlusion.

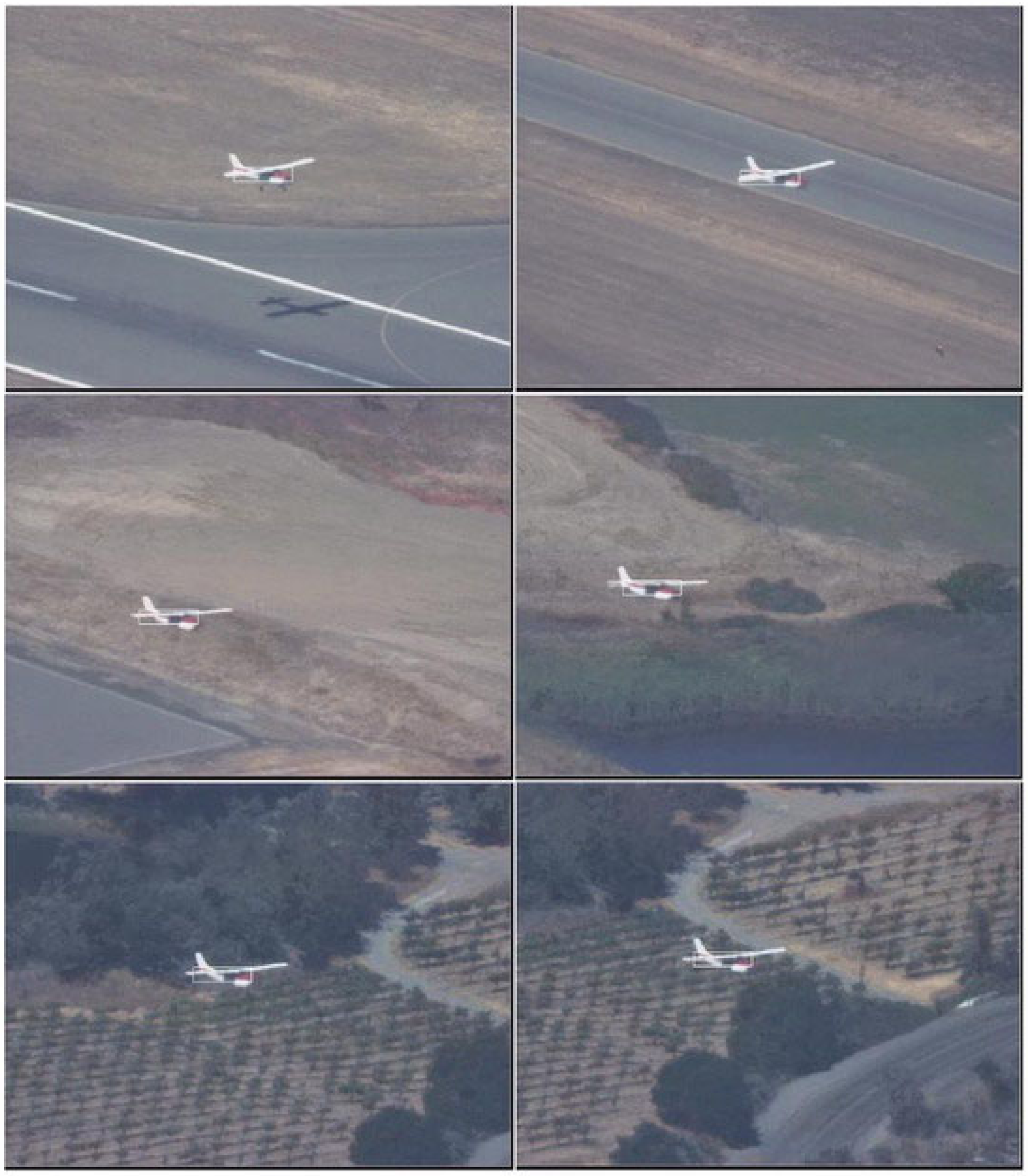


Figure 2.15: Kernel-based tracking algorithm tested on aerial videos from Naval Data Test Set I.

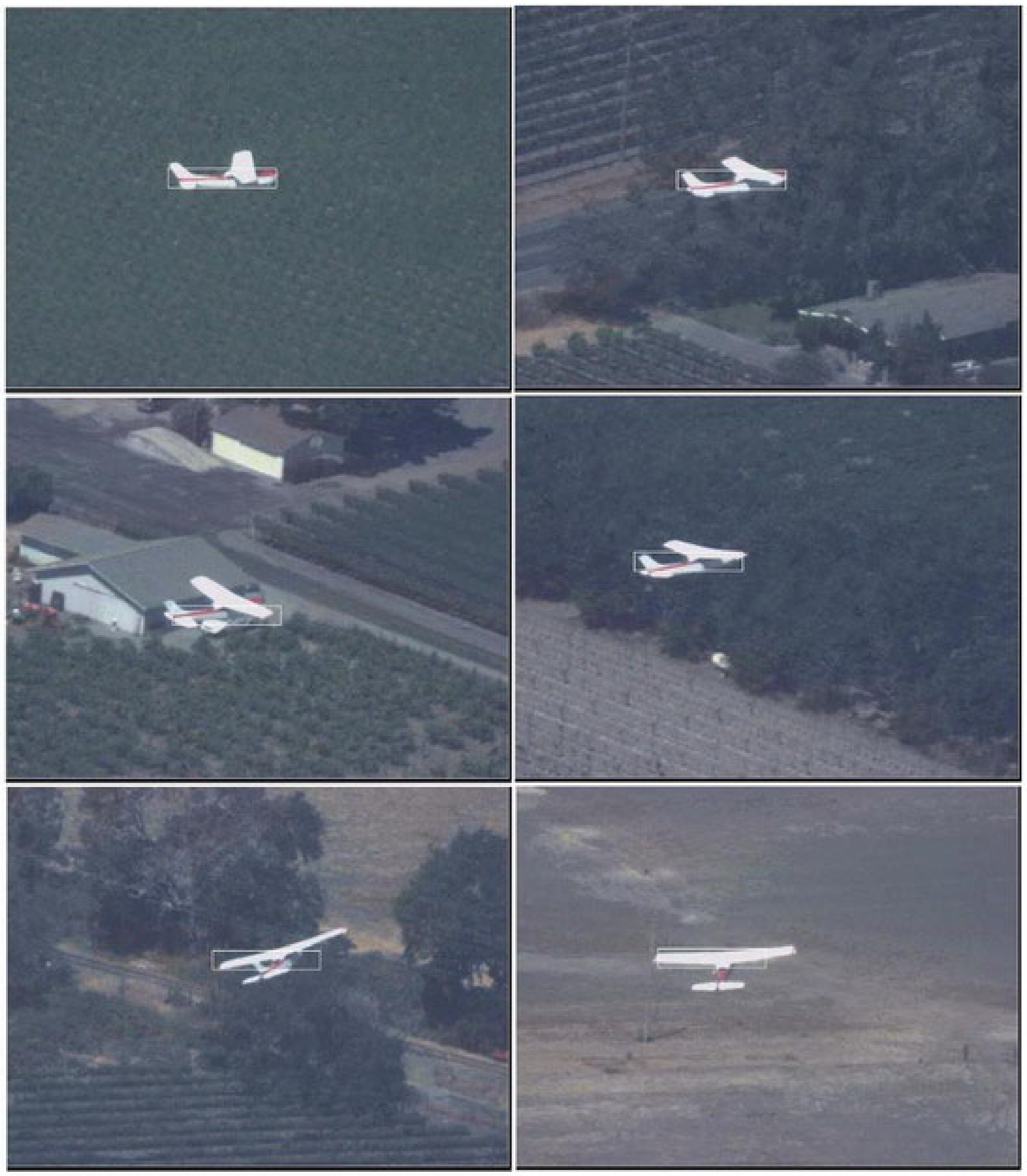


Figure 2.16: Kernel-based tracking algorithm tested on aerial videos from Naval Data Test Set II.

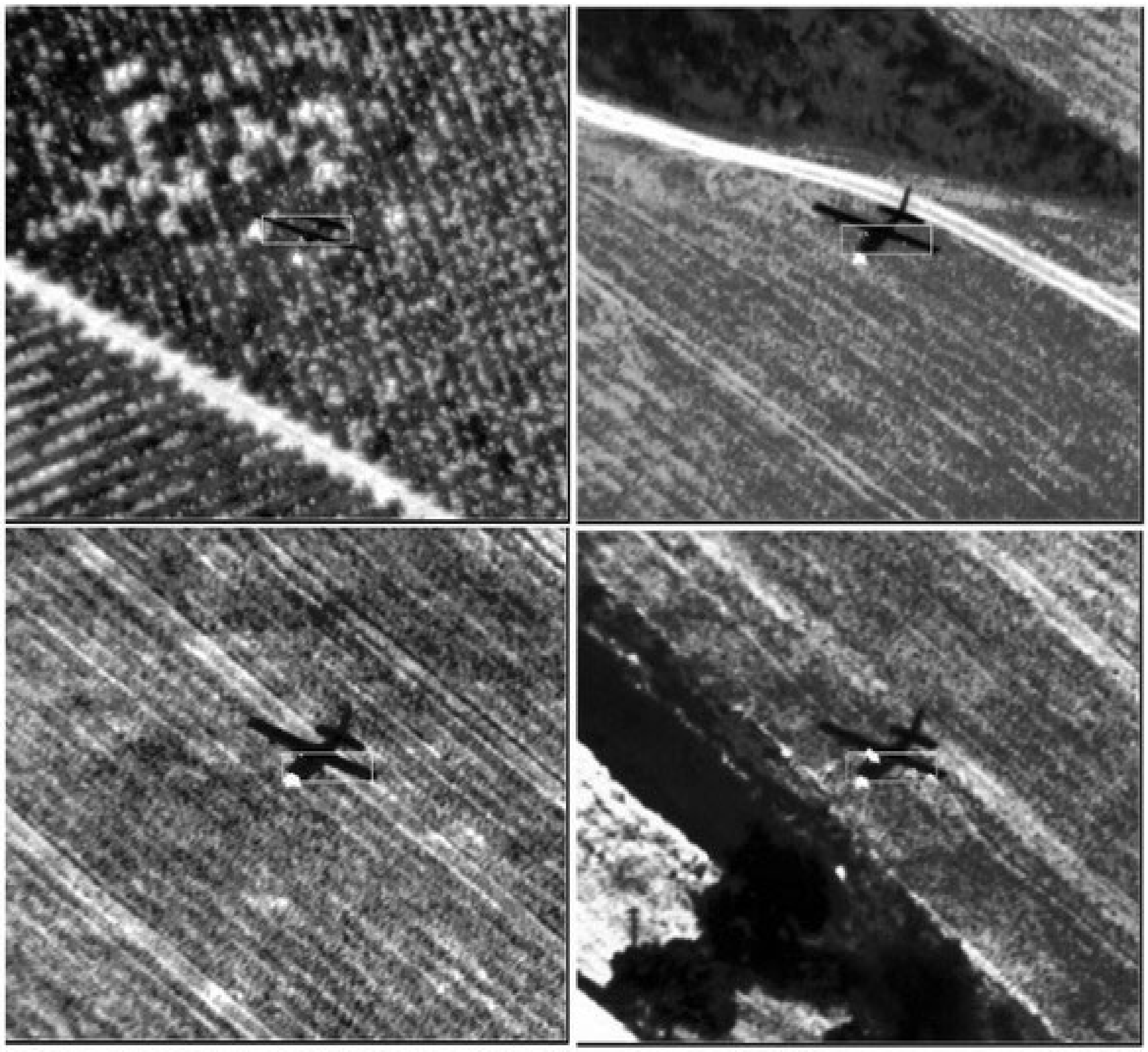


Figure 2.17: Kernel-based tracking algorithm tested on aerial videos from Naval Data Test Set III.

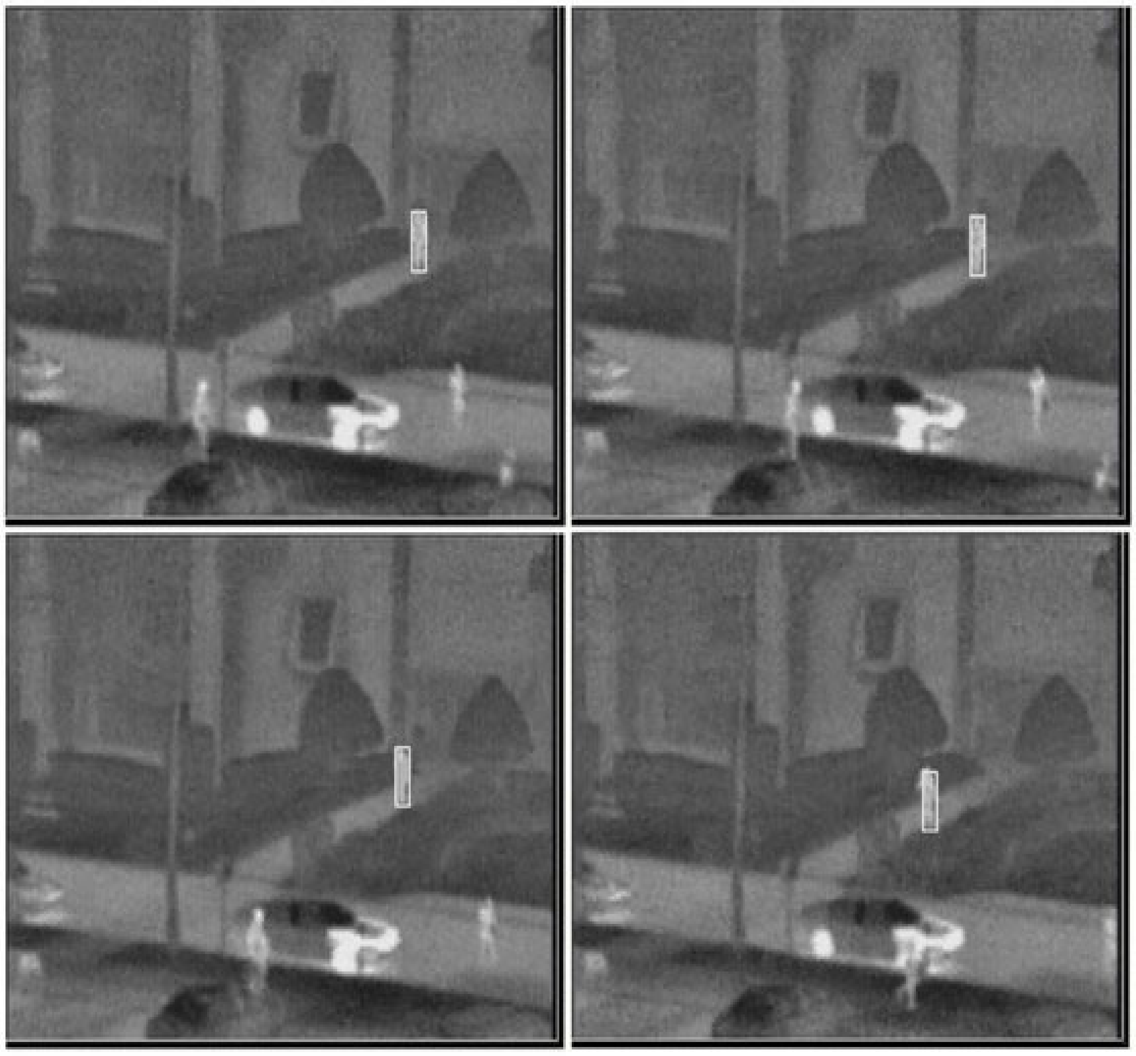


Figure 2.18: Kernel-based tracking algorithm tested on aerial videos from Naval Data Test Set IV.

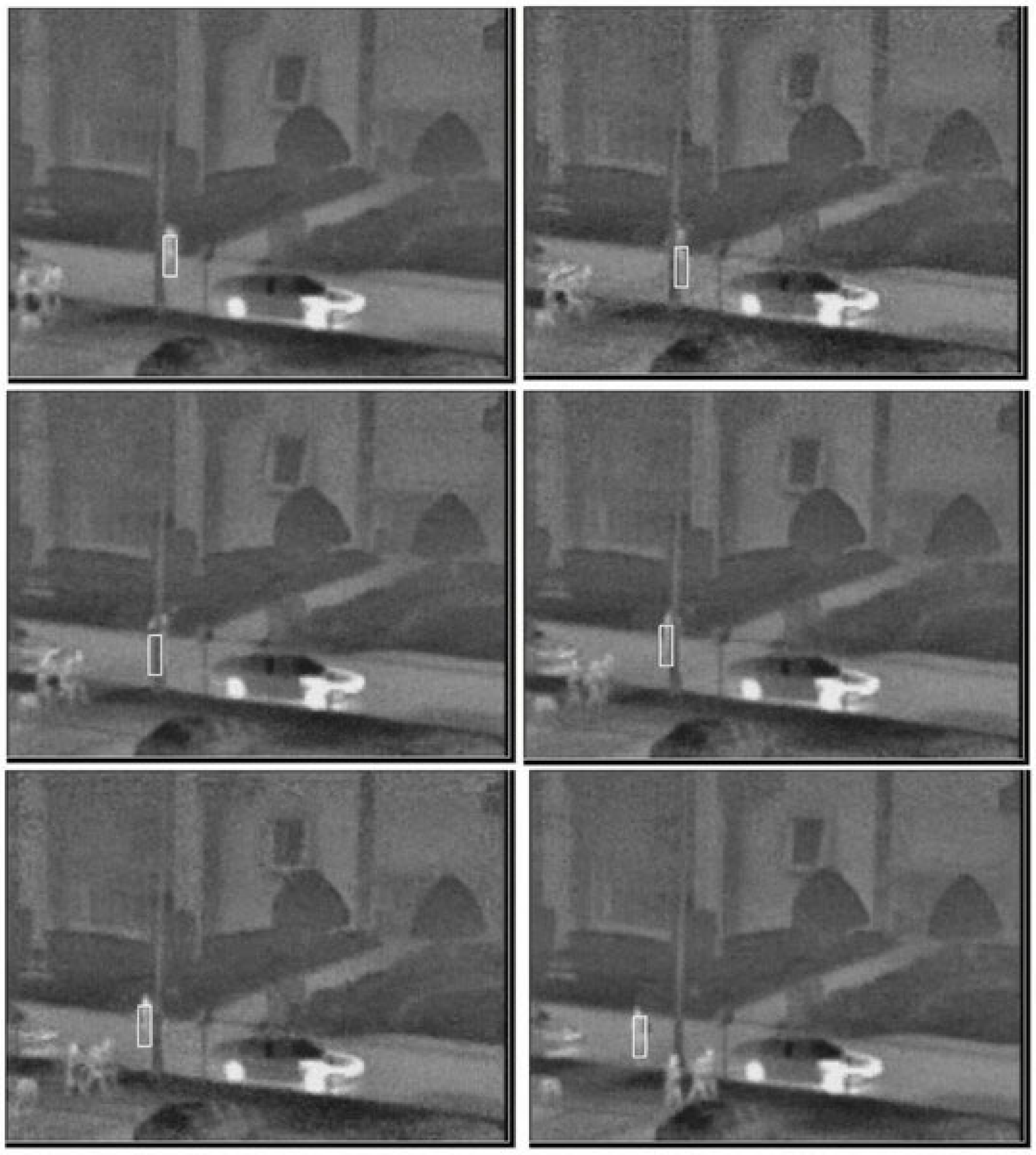


Figure 2.19: Kernel-based tracking algorithm tested on aerial videos from Naval Data Test Set V.

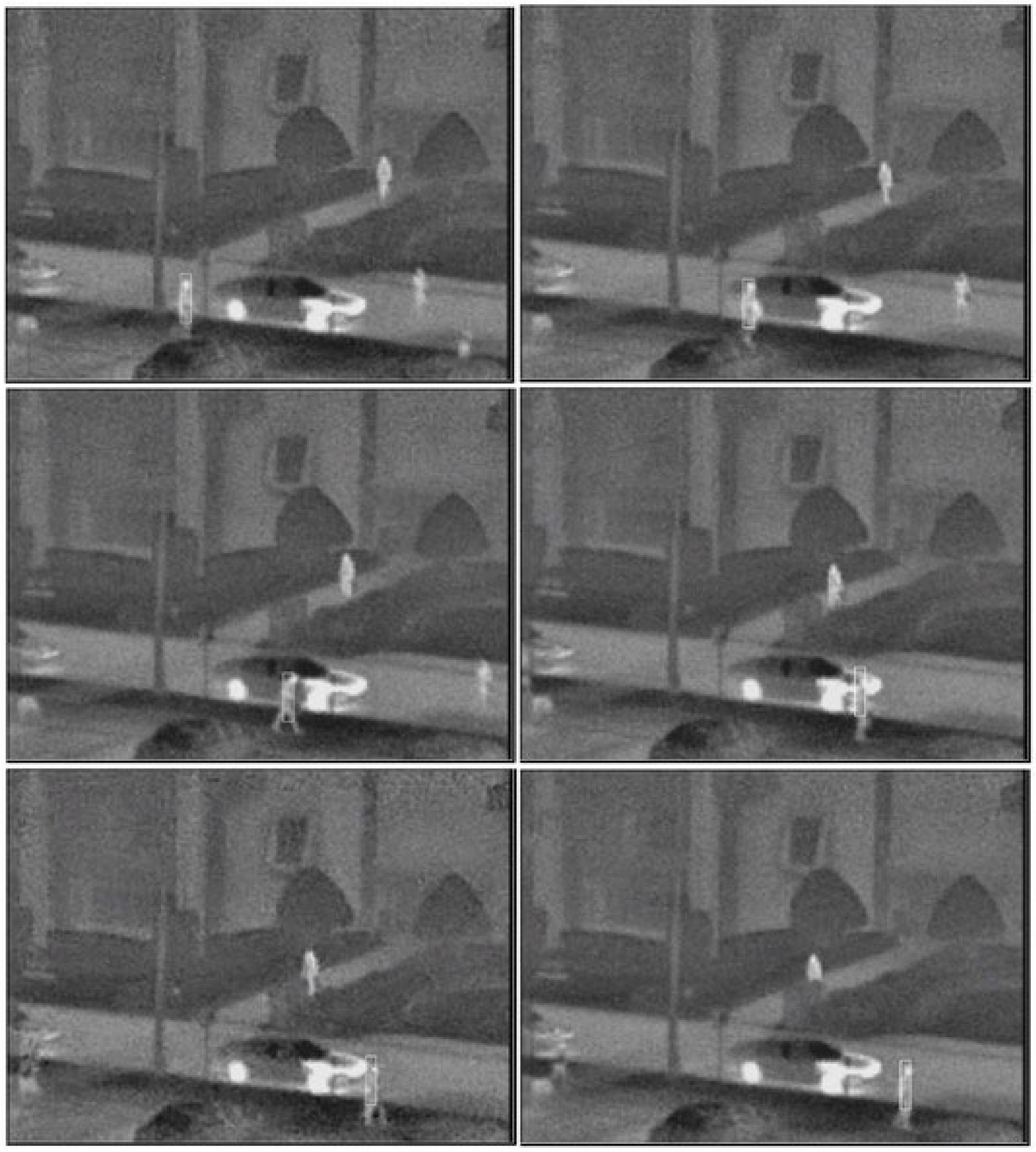


Figure 2.20: Kernel-based tracking algorithm tested on aerial videos from Naval Data Test Set VI.

Chapter 3

Registration Survey

Registration is a task to match two or more pictures taken at different times, from different sensors, or from different viewpoints[86]. A wide range of algorithms has been developed to register various data images. This survey will organize some existing methods and then focus on multi-modal image registration techniques. In particular, information theoretic methods will be detailed such as mutual information based registration, entropy based registration, etc.

3.1 Classification of Registration Methods

3.1.1 Classficication Methods

Registration methods can be classified according to different rules. A registration algorithm usually, without loss of generality, is composed of four components(also called elements)[86]: a feature space, a search space, a search strategy, a similarity metric. Registration algorithms are then often classified or named based on the combination of these components, e.g., Fourier based registration, which is classified by feature space. It is just one or more elements on which some reseachers have presented new ideas and hence improved the existing algorithms.

A finer classfication method[87] presented nine basic criteria. Those criteria are the extension of the four components mentioned in literature [86].The statistics of the classification according to such criteria shows definite trends in the evolving registration techniques[87].

- Dimensionality
- Nature of registration basis
- Nature of transformation
- Domain of transformation
- Interaction
- Optimization procedure
- Modalities Involved
- Subject
- Object

3.1.2 Explanation on Terms

– Dimensionality

Registration problems can be categorized depending on the number of spatial dimensions involved. Table 3.1 shows all sorts of registration categories of different dimensionality. In general, 3-D information is obtained either from image sequences containing stereo information or from 3-D models constructed based on still images.

Table 3.1: Registration Dimensionality.

Modality	spatial dim = 2	spatial dim = 3
2D/2D	only spatial infor is applied	same
2D/3D	image and 3-D Model	-
3D/3D	3-D infor. needs to be constructed	time(t) is the 3^{rd} dimension

– Subject

Subject denotes the number of subjects involved in the images to be registered [88]. It can be either a single one, e.g. the brain images from the same person, or multiple ones, e.g. from different persons. In the earlier literature [87], J. B. Antoine Maintz etc. described a detailed classification of subjects in medical image registration:

- Intersubject: the registration uses two images acquired of different patients.
- Intrasubject: the registration uses two images acquired of a single patient.
- Atlas: the registration uses an image and another from a database.

– Search Space[86], also called Nature of transformation[87]

The search space is generally refered to as the class of transformations from which we would like to find the optimal one to register images[86]. The most common transformations considered in existing methods are [86][87]:

- rigid
- affine
- projective

- perspective
- polynomial

search space in most cases is composed of all possible transformations. To reduce the computational cost however, some assumptions are usually made on transformation classes used in each registration technique. Models can be built to describe the possible transformations and hence to reduce the search space for calculations, especially when the source of misregistration is known or can be exactly modeled. As a rule, they are categorized as allowing global or local transformations[86].

Polynomial transformation is often utilized in global alignment, if other transformations can not account for the distortions or, if there is not enough information available to model the transformation. A global transformation is given by a single equation or a single set of parameters, which will be used to map the entire image. While a local transformation maps one image onto another differently depending on the spatial location. The comparisions of global and local transformations are listed in Table 3.2. This classification corresponds to ”Domain of Transformation” in literature[87].

Recently, rigid and affine transformation are usually applied as global while curved transformation as local[87].

– Domain of Transformation

In[87], domain of transformation is refered to as global or local as shown in Table3.2.

– Feature Space[86], also called nature of registration basis[87]

Feature is the information used for matching images. It can be either manually selected by an expert, or obtained by extracting salient structures. Some correlation methods compute the measures

Table 3.2: Global and local transformation.

	Global	Local
Number of Transformation	one single	multiple
Computational Cost	high	low
Advantage	use the entire image to compute the transformation parameters	use only local info and hence less influenced by other parts

only on some feature points instead of the entire image to reduce the computational cost. In addition to low computational cost, feature points can help correspond regions of physical meaning. Feature selection, to some extent, determines the registration quality as will be described later.

Salient objects are often detected as features in algorithms. We may have to skip the problem of defining "salient objects", as it is still an open problem in computer vision. As a rule, the common features include [89]:

- edges
- contours
- closed-boundary regions
- line intersections
- curves or surfaces
- corner

After being further processed, these features can be represented by representative points, which are called Control Points(CPs) [86][89]. Control points can be either intrinsic or extrinsic. Intrinsic control points are markers unrelevant to the image data itself[86]. They are often placed in the scene and easily identified, serving as reference points. But there are not always available intrinsic control points in some cases. More often than not, control points are generated using image data either manually or automatically, so-called extrinsic. Manual points, i.e., points selected by human interaction, has advantages over automatical ones in that they can impart subjective information. They require knowledgeable people and hence this is not always feasible in case where there is a large amount of data. Therefore, automatical points are often used in some applications. Possible problems include:

- The number of control points. It can be very big, which increases the computational cost and accuracy. Meanwhile, too many points may make matching difficult.
- What features should be taken in specific applications?
- Update control points after each iteration? After control points being detected, they can be matched from a misaligned image to the reference image. Usually, this involves an iteration. Can those control points be updated? Updating control points means more computational cost, nevertheless, if the matching/searching algorithm goes on the right way, i.e. it is registering the misaligned image closer to the reference image, the updation may save some computaional cost of matching.
- Similarity metric is to measure the distance between feature points. It should be taken into consideration together with control(feature) points. Consequently, to improve the performance of registration, not only can we choose feature points, but also, new similarity metric can sometimes register better with the existing features.

It is worth pointing out that **extrinsic** and **intrinsic** are used differently by literature[86] and [87]. In [87], "**extrinsic** registration methods rely on artificial objects attached to the patient... The main drawback of extrinsic registration are the prospective character, i.e., provisions must be made in the pre-acquisition phase, and the often invasive character of the **marker objects**. ...Since extrinsic methods by definition cannot include patient related image information...". " **intrinsic** methods rely on patient generated image content only. Registration can be based on a limited set of identified salient points(landmarks), on the alignment of segmented binary structures(segmentation based), ...”

– Similarity Metric Space

Similarity metric is the ojbect to be optimized by registration. Selecting similarity measure is closely related with selecting matching features as it measures the similarity between these features of the float image and refence image. In fact, selected feature has a heavy influence on selecting similarity metric. Consequently, the two spaces are often considered and selected together with each other. Typical similarity measures are:

- cross-correlation
- sum of absolute difference
- Fourier-invariance properties such as phase correlation
- correlation of edge images

Some existing algorithms required assumptions on features, e.g., control points are easy to identify. However, when those requirements are not met in real applications, more general algorithms are preferable for dealing with various imaging situations so as that similarity metric can be applied in a wide range of features, but not limited on the cases where the assumptions must be satisfied. This idea naturally led to the introduction of mutual information as a registration measures dating back to 1990s[90][91] as discussed in section 3.2.

If gray or intensity values are used as feature, a similarity is usually chosen to be more noise robust. While features rather than intensity are selected, the effects of noise may be reduced to some degree. Of course, this is paid for by the possibility that some valued variations or distortions are not recogzied.

– Optimization

Powell’s multidimensional direction set optimization seems to be the most commonly used method. Stochastic optimization methods have shown their ability in some field, and been applied in registration, e.g. in [90]. Some registration system does not pay too much attention into optimization procedure[92], although it is itself a problem being researched.

3.2 Mutual Information Based Registration Methods

Maximization of mutual information appears to be a powerful registration methods[90]. It is an intensity-based, rather than feature-based method and can date back to 1990s[88]. Essentially, mutual information is a similarity metric between images, and therefore, it can also be viewd as a distance. Three information theoretic metrics are commonly applied as similarity metric in image registration.

- Entropy (read more papers)
- Mutual information
- Kullback relative entropy

3.2.1 Difficulties Encountered by Other Methods

Selection of features often has effect on the selection of similarity metric. And in fact, they together play a key role in registration. From this point of view, selecting features is itself a problem. For example, gradient-based methods or edge-based ones can have difficulties in images lacking discontinuity, since neither edges or gradients are stable properites with respect to lighting changes. Furthermore, how to extract features is still another problem. Even in the absence of noise, for example, the defintion of edge points may be an ill problem, while the presence of noise will make it more difficult to extract edges than we may expect. This becomes prominent especially in image registration, as the features determine the quality in a certain sense.

The advantage of intensity-based methods over feature-based methods lies in the fact that it can be applied to a wide variety of image modalities. This is orignated from the nature of mutual information–it does not need any particular features information, while only intensity information is used. However, mutual information is not panacean. Its drawback comes out of its advantage in the sense that it takes each pixel as equal without taking into spatial information. Some researchers incorporated such information into mutual information based registration system and have improved its performance [92][94].

3.2.2 Implement Mutual Information Based Algorithms

Mutual information-based registration methods can be formulated as [90][91]:

$$\hat{T} = \arg\max_{T} I(T)$$

where

$$I(T) = I(u(x), v(T(x))) \tag{3.1}$$

here x is the coordinates, $u(x)$ and $v(x)$ are two images to be registered, or $v(x)$ is the image to be registered(also called a floating image), while the other $u(x)$ is a model or a fiducial image(also called the reference image). $I(u, v)$ is defined as the mutual information between the two images u and v:

$$I(u, v) = H(u) + H(v) - H(u, v) \tag{3.2}$$

$H(u)$ is the information entropy of u. Notice that u is a random variable denoting the gray value of an image. The distribution of u will be used to calculte its entropy. Moreover, From (3.2), we find that $H(u)$ is a function of not only u explicitly, but also x implicitly, which is carrying spatial information. It is true that different images could have the same entropy if only they are of the same gray distribution while of totally different contents.

Among three basic elements of implementing mutual information based methods are sample selection, probability distribution estimation, and interpolation [88][90][93][96] as discussed below.

Histogram Estimation

To estimate entropy of an image and further mutual information of two images, gray distribution (probability density function: pdf, or probability mass function: pmf) need to be estimated firstly. In literature, gray pdf or pmf is often used interchangeably with histogram [88][93]. For a single image, histogram is estimated by two steps:

1. count the number of times each gray value occur in the image, and
2. divide those numbers by the total number of occurances.

For joint histogram of two images, it can be estimated by similar two steps:

1. Count the number of times each entry occurs
2. Divide each entry in the histogram by the total number of entries. Herein, ‘entry” is refered to as a pair of gray values that corresponding pixels in the two images take on.

Parzen windows has been widely used in approximating histogram. Philippe[93] proposed to use separable Parzen windows and showed that the selection of a Parzen window that satisfies the partition of unity can simplify the estimating problem. It is Parzen windows that formulates the mutual information criterion as a continuous and differentiable function of registration parameters. Also, if marginal probability is calculated based on the estimated joint probability, then it will explicitly depend on transformation T in equation (3.2). By introducing partition of unit constrait, this effect can be avoided[93].

Viola et al.[90] proposed to estimate joint histogram on the basis of Parzen window formulated by superposition of Gaussian density functions:

$$p(z) \approx \frac{1}{N_A} \sum_{z_j \in A} G_\psi(z - z_j) \tag{3.3}$$

This approximation of gray distribution allows for the derivation of estimating the entropy of an image. Moreover, it enables stochastic maximization of mutual information.

Sample Selection

Sample selection is to decide how many and which pixels will be used to estimate histogram. Either all pixels, or a subset, or a superset [95] can be used. In case a superset of coordinates is used, an interpolation method is needed to find the gray values of $u(x)$ at coordinates that do not coincide with any coordinate of $v(x)$. In case a subset is used, a sub-sampling method is needed to sample images, and thus to increase speed. It should be noted that some continuity of gray-values has been assumed in mutual information based registration methods.

Interpolation

Jeffrey Tsao[96] think of mutual information based registration methods involve three steps: 1) Estimate histogram, 2) Determine similarity metric, 3) Optimize the similarity metric with respect to parameter T in [90]. Another technique usually involved in registration methods is interpolation algorithm, which is needed to estimate the gray values at nongrid positions, whenever the voxel grids of two images are not exactly aligned. The registration accuracy varies with interpolators. Jeffrey Tsao[96] characterize the artifacts from the following eight interpolators and investgates efficient strategies to overcome these artifacts:

- nearest neighbour
- linear
- cubic Catmull-Rom
- Hamming-windowed sinc
- partial volume
- NN with jittered sampling(JIT)
- NN with histogram blurring(BLUR)

- NN with JIT and BLUR

In interpolation, nonidealities(e.g. noise) introduce errors to the interpolated voxel intensities. To minimize these errors, data-consitent interpolator is often desired, so that the gray values at grid positions do not change after interpolation. As a result, errors are only introduced at non-grid positions during interpolation. This implies the amount of interpolation errors will vary with the extent of interpolation.The variation may result in fluctuations in similarity metric and hence affect registration accuracy. Jeffrey Tsao[96] found some methods are susceptible to such fluctuations. Therefore, the selection of an interpolator for the purpose of image registration should take into consideration not only the closeness of the interpolated image to the original one, but also the effect of interpolation on similarity metric.

Interpolation, is not a basic element of registration methods, but has a significant effect on registration accuracy [96][98]. Even it may cause misregistration for the local extrema without interpolation or one not appropriate to similarity metric, mutual information. Also, interpolation may be applied a great number of times during the registration process, therefore, it necessitates a tradeoff between accuracy and speed. However, the task of yielding a final registered image requires interpolation only once. Consequently, another interpolation may be more appropriate for this task.

3.2.3 What Mutual Information Cannot Do

Mutual information serving as a good metric, and its capability in multimodal image registration, rely on the assumption that regions of similar structures in one image would correspond to regions in the other image consisting of similar structures. A lot of information theoretic algorithms are based on only intensity information [88][90][91] [93][94][95]. This may bring the following problems.

- Mutual information based methods, at least for the first glance, just make use of intensity information. However, Images may contain meaningful objects which help register.

- Mutual information in image registration is a distance metric. Besides its meaning in information theory, are there any physical meaning or geometrical meaning or else others? Section 3.4 will discuss the derivation from ML to MI.

- New similarity metrics seems to have been able to make existing algrorithms adapted to more wide range of registration problems, which almost always comes with new features. From this point of view, we can try to propose new algorithms by exploring new features.

  - Intensity was an existing feature, but most intensity-based methods are limited on special kinds of problems.

  - Content based, as mentioned before, may help even just simpler extraction methods are used.

3.3 Maximum Likelihood Based Registration Methods

Maximum Likelihood(ML) based registration methods have the same ability as Mutual information(MI) based ones to register images from different imaging modalities.

3.3.1 A Multi-modal Image Registration Method

The underlying assumption is that the intensities in two images to be registered are probabilistically related regardless of whether they are imaged from a single modality or multiple modalities. Based on this assumption, two images are considered and defined to be registered when the likelihood reaches its maximum value. To be consitent with nomenclature as used in section 3.2.2, ML can be formulated as follows:

$$\hat{T} = \arg\max_{T} L(T) \tag{3.4}$$

$$L(T) = L(u(x), v(T(x))) \tag{3.5}$$

Comparing (3.2) and (3.5) provides the insight that Both MI and ML are themselves similarity metrics. In other words, Both of them improve registration performance, e.g., accuracy, by applying new similarity metrics. In particular, they work better than others in multi-modal image registration, mostly because, the similarity metrics are more robust than others in dealing with multimodal image registration.

Similar to MI metric[99], ML is also a full volume-based method since it does not require feature extraction and relies on intensity information only. Also, both of them need to estimate probability relation between two images.

In contrast to MI or other entropy techniques which have a lower bound but a variable bound that is image content (or image data) dependent, ML has a lower bound of 0 and an upper bound of 1. However, none of [99][100][101] has discussed this advantage, if it is of.

3.3.2 Implemention of ML

Expression for ML

Assume two images $u(x)$ and $v(x)$ with gray values ($u_1, u_2, \ldots, u_m$) and ($v_1, v_2, \ldots, v_n$) respectively. A conditional probability density of $w_{ij} = p_{v|u}(v_j|u_i)$ can be constructed either by knowledge-based or self-consitent techniques[99]. Then, $L(T)$ in equation (3.5) can be written as

$$L_{v|u} = \prod_{x \in R} w_{v(x)|u(x)} \tag{3.6}$$

Where R is the overlapping region of the two images $u(x)$ and $v(x)$. The number of entries in R may vary with registration parameter T, and sometimes R can be a subset of the overlapping region. Consequently, the likelihood defined in (3.6) relies on the number of entries. To avoid this, a normalized likelihood is often used

$$L_{v|u} = \frac{1}{N} \prod_{x \in R} w_{v(x)|u(x)} \tag{3.7}$$

where N is the number of entries.

Notice that $w_{ij} < 1$ and for the convenience of computation, a logarithmic likelihood is often applied as in other ML estimation algorithms

$$L_{v|u} = \frac{1}{N} \sum_{x \in R} \log \left(w_{v(x)|u(x)} \right) \tag{3.8}$$

Conditional pdf Estimation

One indispendible technique for MI and ML is intensity pdf estimation. MI methods estimate joint pdf, while ML estimate conditional pdf.

One estimation method is so called “Knowledge-Based”. This technique requires some pairs of images that have already been registered. One may get the statistics on the transition probabilities according to registered images. This method can be extended to a general case where the the statistics is stationary in the sense that it does not vary with different images. In such cases, statistics can be obtained from any well-registered images in advance. The probable problem here is whether the statistics is stationary. However, it has been assumed whenever this method is used. This may be especially useful if stationarity holds in some multi-modal registration problems.

Another method is so called "Self-Consistent" in that the overlapping region in current registration is used to calculate the transition probability used in calculating the likelihood in next registration[99].

3.3.3 What is Left for ML

Are there any relations between MI and ML? ML needs to estimate conditional pdf, and MI joint pdf. Conditional pdf is coherently related with joint pdf, so there may also be some coherent relations between MI and ML in terms of similarity metric, performance.

Is it similar to MI in the sense that conditional pdf estimation plays a key role and hence interpolation is very important? Can we conduct some experements to verify the assumption underlying ML, especially in multi-modal registration techniques?

3.4 Utilize Spatial Information

A drawback of mutual information is that the dependence of the gray values of neighboring voxels is ignored, since the Shannon entropy is only determined by pdf of an image and joint pdf of the two images. The condition of independence of gray intensities, although often assumed, does not hold in general[88][92][94]. Spatial information on how gray intensities distribute across an image has no effect on the calculation or estimation of pdf. Two images of the same pdf can be totally different images in terms of content, imaging modes, etc.

3.4.1 Combine Multual Information with Gradient Information

Josien et al. proposed imparting gradient information as spatial information[94]. Assume the image to be registered and the reference one will be prefiltered before registration. Let $G_\sigma(\mathbf{x})$ denote a gaussian kernel of scale σ (standard deviation). Then, the gradients of a gaussian-filtered image can be calculated:

$$\dot{u}(\mathbf{x}) = \nabla\{u(\mathbf{x}) \star G_\sigma(\mathbf{x})\} = u(\mathbf{x}) \star \nabla G_\sigma(\mathbf{x}) \tag{3.9}$$

$$\dot{v}(\mathbf{y}) = \nabla\{u(\mathbf{y}) \star G_\sigma(\mathbf{y})\} = u(\mathbf{y}) \star \nabla G_\sigma(\mathbf{y}) \tag{3.10}$$

In [94], the gradients are calculated by convolving an image and the first derivative of Gaussian kernel. After this, the angle between the gradient vectors at correponding pixels **x** and **y** is defined

$$\alpha(\mathbf{x}, \mathbf{y}, \sigma) = \arccos \frac{\dot{u}(\mathbf{x}) \cdot \dot{v}(\mathbf{y})}{|\dot{u}(\mathbf{x})| \cdot |\dot{v}(\mathbf{y})|} \tag{3.11}$$

The gradients of the images from different modalities can point different directions but with the same orientation while they are well registered. In view of this, [94] suggested using an angle metric favoring the gradients with the same orientation

$$w(\alpha) = \frac{\cos(2\alpha) + 1}{2} \tag{3.12}$$

The final similarity metric is the product of mutual information and gradient vectors

$$M(U,V) = G(U,V)I(U,V) \tag{3.13}$$

$$G(U,V) = \sum_{(\mathbf{x},\mathbf{y})\in(U\cap V)} w(\alpha(\mathbf{x},\mathbf{y},\sigma)) \cdot \min(|\dot{u}(\mathbf{x})|, |\dot{v}(\mathbf{y})|) \tag{3.14}$$

Equation (3.14) serves as the final similarity metric including spatial information. Mutiplication is prefered to addition, as addition of two terms requires normalization[94]. Both magnitude and direction of gradient vectors contribute to the similarity metric. So the spatial information, i.e. gradient vector in this case, defines four metrics which will effect the final registration quality. Modification on these four metrics will improve the performance of the whole registration algorithm.

- The angle between two gradient vectors at corresponding pixels in equation (3.12)
- The multiplication in equation (3.13)
- The “min” and product of angle and magnitude in equation (3.14)
- The summation over the whole overlapping region in equation (3.14)

3.4.2 Formulate Spatial Information

Mert et al. started utilizing spatial information from maximum likelihood [92]. Define a function $s(\mathbf{x})$, then Mert developed a ML framework with additive Gaussian noise as follows

$$P_{V|U,T}(V = v|U = u, T = t) = \prod_{\mathbf{x}\in U\cap V} P_{W(\mathbf{x})}[w(\mathbf{x}) = u(\mathbf{x}) - g(v(t(\mathbf{x})), s(\mathbf{x}))] \tag{3.15}$$

As a rule, taking logarithm to both sides of the above equation gives:

$$m(t) = \sum_{\mathbf{x}\in U\cap V} \log P_{W(\mathbf{x})}[w(\mathbf{x}) = u(\mathbf{x}) - g(v(t(\mathbf{x})), s(\mathbf{x}))] \tag{3.16}$$

The amazing result for (3.16) is that Mert proved the expectation of $m(t)$ is equal to a certain conditional entropy

$$E[m(t)] = -NH_{U|V,S}(U(\mathbf{x})|V(\mathbf{x}), S(\mathbf{x})) \tag{3.17}$$

This may be the first step to build relationship between MI and ML.

One point worth noting is the function $s(\mathbf{x})$ appearing in equation (3.15) and (3.16) incorporates spatial information into information theoretic algorithms. This technique indeed, applies not only in such algorithms, but also in any other algorithms that need spatial information. Detailedly, it is implemented by function $g(v(t(\mathbf{x})), s(\mathbf{x}))$. However, [92] did not give any idea on how to construct application specific $g(\cdot)$.

Another point is the difference which lies between the definitions of ML in equation (3.15) and (3.6). The former assumed additive Gaussian noise model, while the latter applies non-parametric(sample based) technique to estimate pdf of an image or joint pdf. Sample based techniques have been applied in many fields, e.g. example-based facial sketch generation[102]. For non-parametric registration methods, there has not been papers discussing incorporating spatial information into estimating (conditional) pdf in equation (3.6).

3.4.3 Spatial Information in Mutual Information Based Algorithms

Two ideas can be further developed for algorithms in [92]. One is to build the relationship between ML and MI. It, if built under some constraits, can provide underlying basis for MI serving as a similarity metric. Meanwhile, all methods applied together with ML imparting spatial information can be applied with MI.

$s(\mathbf{x})$ is the function carrying spatial information. Content based registration algorithms modify MI by setting $s(\mathbf{x})$ a prior. On light of the fact that $s(\mathbf{x})$ works essentially by adding useful information, the problem of pluging spatial information into MI can be extended to

- Segmentation or content based registration

- Application specific registration

- Human guided(aided) registration

Parametric models usually represent kinds of deformation by a moderate number of parameters[104]. It is customary to apply parametric models into those deformation problems, where they can be well represented by only a small number of parameters[103]. Examples include spline, wavelet , etc. The drawback of the parametric methods is the restriction of the allowed deformation space. Nonparametric methods on the contrary, is totally unconstricted regarding the deformation space. However, despite this advantage, there seems to be more room for nonparametric methods. Only a limited number of papers applied nonparametric methods in registration.

Chapter 4

Implementation and Modification of Existing Algorithms

4.1 Software Specifics

The software includes those parts similar to what are described above comprising mutual information based registration algorithms.

- I/O. Read and/or Write image files from and/or to storage media.
- Interpolation. Linear interpolation is utilized at this point.
- Optimization. Modified Powell direction set is applied.
- Pdf(Probability Density Function) Estimation. At this point, (joint)normalized histogram is simply used as (joint) pdf.
- Entropy calculation.

4.1.1 Developing/Testing Environment Specifics

All source code was programmed in C/C++ on Linux.

- Image size: 640×480
- Image format: 256-level gray images
- Operating System: Linux version 2.4.21-20.EL (Red Hat Linux 3.2.3-42)
- Compiler: gcc version 3.2.3 20030502

4.1.2 Problems Encountered in Testing Data

Optimization is an indispensible element in registration and hence it will definitely effect the registration quality. But, optimization is not all of the problems registration is supposed to solve. Many existing algorithms choose a certain optimization algorithm according to their applications. Or, they just choose the popular method, e.g. Powell routine, to optimize their objective functions[88]. Powell method is able to guarantee the convengence of quadratic forms [105][106], while, it says nothing on convegence of general multiple dimension optimization problems.

Powell's method has been applied in testing images. It is composed of two parts, the first one conjugate direction search, the second one line minimization along one direction. Line minimization calls a routine program for initially bracketing a minimum, which is denoted as

$$\begin{aligned} void \quad mnbrak(&double*ax, double*bx, double*cx,\\ &double*fa, double*fb, double*fc,\\ &double(*func)(double)) \end{aligned}$$

This routine can bracket the minimum of a function into a region $[ax, cx]$. Experimental result shows it works well in some "ideal" cases, but not in the data used in this report.

Table 4.1: Real optimization process.

number of pixels in overlapping area	mutual information
307200	0.000000
306720	0.248788
126483	0.305751
5001	0.998649
1017	1.645018
640	3.666301

Table 4.1 shows the number of pixels in the overlapping area between a reference image and a float image, and the corresponding mutual information. The search space is chosen to be affine transformation with initial values set to 0. However, this optimization does not account for the difference between the two images. In fact, the optimization process has not been on the right track from the beginning in this particular case. The reason is that the bracketing function $mnbrak$ has not worked as it is expected. In other words, it often brackets the local minimum instead of the global one into a certain region along a direction.

4.2 Modified Powell's Method

As mentioned in section 4.1.2, the line minimization does not work well partially because $mnbrak$ does not work as expected. At this point, two directions have been tried to fix up such a problem. The underlying idea is either try to make $mnbrak$ work or otherwise give it up.

4.2.1 Optimize with a Meaningful Initial Value

To make Powell work, [95] assumes we are able to select an initial value inside the atraction pool of the minimum. Stimulated by this idea, we can also choose a "good" initial value knowing $a\,priori$ information. We tested this method on two consective frames of a video sequence and the result is shown in table 4.2. We can see from the table that a "good" initial value does good for our line optimization.

Table 4.2: Optimization with meaningful initial value.

number of pixels in overlapping area	mutual information
306081	2.519544
153280	0.178674
479	2.590159
306560	2.640276
306082	2.773243
306081	2.870510

4.2.2 Exhaustive Search of Line Minimization

The alternative way is to give up $mnbrak$. A natural modification is exhaustive search as applied in literature[92]. Here, we still utilize Powell's frame of conjugate direction search but substitute brent method for a exhaustive search, Please refer to [106] for brent method. Unfortunately, It still can not guarantee the convergence of Powell's algorithm. Viewing this point, a good initial value is preset based on $a\ priori$ knowledge. We tested this modified Powell's method by the following steps:

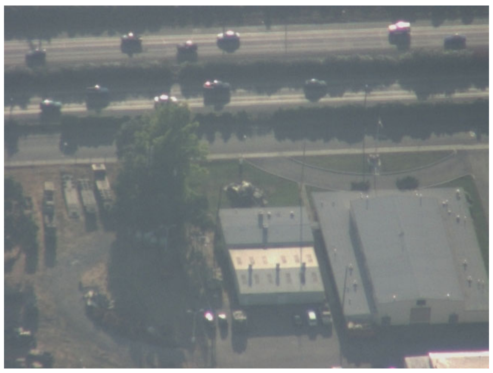
- Rotate an image by a certain angle, e.g., $\pi/6$, and set it as a float image.
- Set the rotation angle of initial affine transformation to $\pi/5.8$.
- Set the translation of initial affine transformation to 0.
- Apply Powell's method with exhaustive line search.
- End optimization when the increase of mutual information is sufficiently small.

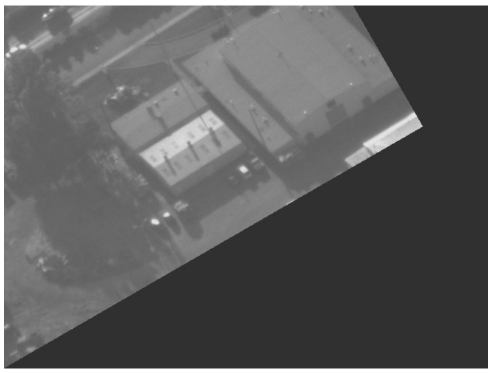
Table 4.3 shows the optimization process with the above steps. The final transformation parameter is shown as affine matrix in equation 4.1. Fig. 4.1(a) shows the reference image to which Fig. 4.1(b) is registered and the registration result is shown in Fig. 4.1(c).

$$\mathbf{A} = \begin{pmatrix} 0.866712 & -0.495409 & -1.647553 \\ 0.505554 & 0.856857 & -1.002103 \end{pmatrix} \tag{4.1}$$

Exhaustive line search can not make sure Powell's method converges to the global minimum, but it is able to avoid the problem $mnbrak$ could encounter that wrong bracketing may mislead the entire optimization process.

At this point, there are two problems left on optimization. One is how to choose initial value by $a\ priori$ knowledge, the other one is "exhaustive search". Since the notorious difficulty of the former, we come to focus on the latter. The line minimization along a direction can be viewed as a one-dim problem. Unfortunately, none of the deterministic existing methods can guarantee the convergence to the global minimum if we know nothing on mutual information as a function of affine transformation parameters. From this point of view, "exhaustive" search is not theoretically feasible at all. Researchers have tried formulating mutual information a continuous(or furthur more, a sufficiently smooth) function, which would allow for the "exhaustive" search [93].


(a) The reference image


(b) The float image

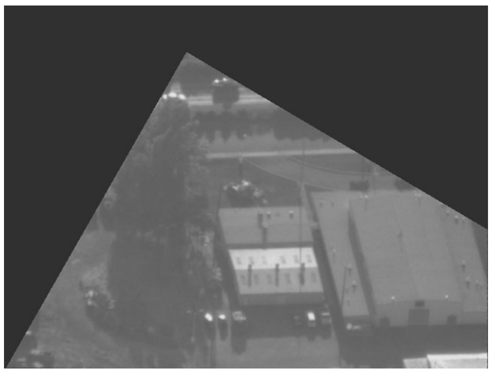

(c) The registered image

Figure 4.1: Registration result

Table 4.3: Optimization with exhaustive line search.

number of pixels in overlapping area	mutual information
22871	0.520090
62302	0.392120
114015	0.353082
171202	0.662163
169064	1.746670
170054	2.482787

Bibliography

[1] Christopher R. Wren, Ali Azarbayejani, Trevor Darrell, Alex Pentland ”Pfinder: Real-Time Tracking of the Human Body”, in: SPIE Photonics East 1995, Vol. 2615 pp. 89-98

[2] Chris Stauffer and W.E.L Grimson.”Adaptive background mixture models for real-time tracking”, CVPR99, Fort Colins.

[3] W. Zajdel, B.J.A. Krise. ”Gaussian Mixture Model for Multi-sensor Tracking.” In Belgium-Netherlands Conf. on Artificial Intelligence, (BNAIC). pp. 371-378. Nijmegen, NL, 2003.

[4] Javed O, Rasheed Z, Shafique K and Shah M, ”Tracking Across Multiple Cameras With Disjoint Views”, The Ninth IEEE International Conference on Computer Vision, Nice, France, 2003.

[5] Stan Birchfield, ”Elliptical Head Tracking Using Intensity Gradients and Color Histograms”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, California, pages 232-237, June 1998

[6] Stan Birchfield, ”Parametric Tracking of Human Contour by Exploiting Intelligent Edge”, Sixth IEEE International Conference on Automatic Face and Gesture Recognition May 17 - 19, 2004 Seoul, Korea

[7] Michael Isard, Andrew Blake, ”Condensation Tracking By Stochastic Propagation of Conditional Density”, 1996, ECCV.

[8] S.C. Zhu and A. Yuille. Region competition: unifying snakes, region growing, and Bayes MDL for multiband image segmentation.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 18(9):884-900, 1996.

[9] N. Paragios and R. Deriche. Geodesic Active Regions and Level Set Methods for Supervised Texture Segmentation.” Int. Journal of Computer Vision, 46(3):223-247, 2002.

[10] A. Elgammal, R. Duraiswami, D. Harwood, and L. Davis. Background and Foreground Modeling Using Nonparametric Kernel Density Estimation for Visual Surveillance.” Proc. of IEEE, 90(7), 2002.

[11] D. Comaniciu, v. Ramesh, and P. Meer. Kernel-Based object tracking.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 25:564-575, 2003.

[12] P. Fieguth and D. Terzopoulos. ”Color-based tracking of heads and other mobile objects at video frame rates.” In IEEE Conf. on Computer Vision and Pattern Recognition, pp. 21-27, 1997.

[13] S. Avidan. Support Vector Tracking.” In IEEE Conf. on Computer Vision and Pattern Recognition, 2001.

[14] M. Black and A. Jepson. EigenTracking: Robust Matching and Tracking of Articulated Objects Using a View-Based Representation.” Int. Journal of Computer Vision, 26(1):63-84, 1998.

[15] C.J. Veenman, M.J. Reinders, and E. Backer.” Resolving Motion Correspondence for Densely Moving Points.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 23(1):54-72, 2001.

[16] K. Shafique and M. Shah. A Non-Iterative Greedy Algorithm for Multi-frame Point Correspondence.” In IEEE Int. Conf. on Computer Vision, 2003.

[17] I. Haritaoglu, D. Harwood, and L.S. Davis. ”W4: real-time surveillance of people and their activities.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 22(8):809-830, 2000.

[18] D. Cremers and C. Schnorr. Statistical shape knowledge in variational motion segmentation.” Image and Vision Computing Journal, 21:77-86, 2003.

[19] C. Stauffer and W.E.L. Grimson. ”Learning patterns of activity using real time tracking.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 22(8):747-767, 2000.

[20] J. Rittscher, J. Kato, S. Joga, and A. Blake. ”A Probabilistic Background Model for Tracking.” In Europen Conf. on Computer Vision, volume 2, pp. 336-350, 2000.

[21] Bjoern Stenger, Visvanathan Ramesh, Nikos Paragios, Frans Coetzee, and Joachim M. Buhmann. ”Topology Free Hidden Markov Models: Application to Background Modeling.” In IEEE Int. Conf. on Computer Vision, 2001.

[22] N.M. Oliver, B. Rosario, and A. Pentland. ”A bayesian computer vision system for modeling human interactions.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 22(8):83-843, 2000.

[23] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh. ”Background Modeling and Subtraction of Dynamic Scenes.” In IEEE Int. Conf. on Computer Vision, 2003.

[24] J. Zhong and S. Sclaroff. ”Segmenting Foreground Objects from a Dynamic Textured Background via a Robust Kalman Filter.” In IEEE Int. Conf. on Computer Visio, 2003.

[25] R.T. Collins, A.J. Lipton, H. Fujiyoshi, and T. Kanade. ”Algorithms for cooperative multisensor surveillance.” Proceedings of IEEE, (10):1456-1477, 2001.

[26] T. Kanade, R. Collins, A. Lipton, P. Burt, and L. Wixson. ”Advances in Cooperative Multi-Sensor Video Surveillance.” In Darpa IU Workshop, pp. 324, 1998.

[27] S. Rowe and A. Blake. ”Statistical Mosaics For Tracking.” Image and Vision Computing Journal, 14:549-564, 1996.

[28] M. Irani and P. Anandan. ”Video Indexing Based on Mosaic Representations.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 20(6):577-589, 1998.

[29] J. Shi and J. Malik. ”Normalized cuts and image segmentation.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 22(8):82-905, 2000.

[30] D. Comaniciu and P. Meer. ”Mean Shift: A Robust Approach toward Feature Space Analysis,.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 24(5):603-619, 2002.

[31] N. Xu and N. Ahuja. ”Object contour tracking using graph cuts based active contours.” In IEEE Int. Conf. on Image Processing, 2002.

[32] M. Kass, A. Witkin, and D. Terzopoulos. ”Snakes: active contour models.” Int. Journal of Computer Vision, 1:321-332, 1988.

[33] V. Caselles, R. Kimmel, and G. Sapiro. ”Geodesic active contours.” In IEEE Int. Conf. on Computer Vision, pp. 694-699, 1995.

[34] S.C. Zhu and A. Yuille. ”Region competition: unifying snakes, region growing, and Bayes/MDL for multiband image segmentation.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 18(9):884-900, 1996.

[35] A. Yilmaz, X. Li, and M. Shah. ”Object tracking using level sets.” In Asian Conf. on Computer Vision, 2004.

[36] R. Ronfard. ”Region based strategies for active contour models.” Int. Journal of Computer Vision, 13(2):229-251, 1994.

[37] N. Paragios and R. Deriche. ”Geodesic Active Regions and Level Set Methods for Supervised Texture Segmentation.” Int. Journal of Computer Vision, 46(3):223-247, 2002.

[38] V. Caselles, R. Kimmel, and G. Sapiro. ”Geodesic active contours.” In IEEE Int. Conf. on Computer Vision, pp. 694-699, 1995.

[39] N. Paragios and R. Deriche. ”Geodesic active contours and level sets for the detection and tracking of moving objects.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 22(3):266-280, 2000.

[40] H.P. Moravec. ”Visual mapping by a robot rover.” In Proc. of IJCAI, pp. 598-600, 1979.

[41] C.G. Harris and M. Stephens. ”A combined corner and edge detector.” In 4th Alvey Vision Conference, pp. 147-151, 1988.

[42] J. Shi and C. Tomasi. ”Good features to track.” In IEEE Conf. on Computer Vision and Pattern Recognition, pp. 593-600, 1994.

[43] S.M. Smith and J.M. Brady. ”SUSAN - a new approach to low level image processing.” Int. Journal of Computer Vision, 23(1):45-78, 1997.

[44] K. Mikolajczyk and C. Schmid. ”An affine invariant interest point detector.” In Europen Conf. on Computer Vision, volume 1, pp. 128-142, 2002.

[45] C.J. Veenman, M.J. Reinders, and E. Backer. ”Resolving Motion Correspondence for Densely Moving Points.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 23(1):54-72, 2001.

[46] I.K. Sethi and R. Jain. ”Finding trajectories of feature points in a monocular image sequence.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 9(1):56-73, 1987.

[47] V. Salari and I. K. Sethi. ”Feature point correspondence in the presence of occlusion.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 12(1):87-91, 1990.

[48] K. Rangarajan and M. Shah. ”Establishing motion correspondence.” Computer Vision, Graphics and Image Processing, 54(1):56-73, 1991.

[49] S. Intille, J. Davis, and A. Bobick. ”Real-time closed-world tracking.” In IEEE Conf. on Computer Vision and Pattern Recognition, pp. 697-703, 1997.

[50] K. Shafique and M. Shah. ”A Non-Iterative Greedy Algorithm for Multi-frame Point Correspondence.” In IEEE Int. Conf. on Computer Vision, 2003.

[51] T.J. Broida and R. Chellappa. ”Estimation of Object Motion Parameters from a Sequence of Noisy Images.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 8(1), 1986.

[52] D. Beymer and K. Konolige. ”Real-time tracking of multiple people using continuous detection.” In ICCV: Frame-Rate Workshop, 1999.

[53] R. Rosales and S. Sclaroff. ”3D trajectory recovery for tracking multiple objects and trajectory guided recognition of actions.” In IEEE Conf. on Computer Vision and Pattern Recognition, 1999.

[54] Kalman Filtering source code. http://www.ai.mit.edu/ murphyk/ Software/index.html.

[55] H. Tanizaki. ”Non-Gaussian State-Space Modeling of Nonstationary Time Series.” Journal of the American Statistical Association, 82:1032-1063, 1987.

[56] Y. L. Chang and J. K. Aggarwal. ”3d structure reconstruction from an ego motion sequence using statistical estimation and detection theory.” In Workshop on Visual Motion, 1991.

[57] C. Rasmussen and G. Hager. ”Probabilistic Data association methods for tracking complex visual objects.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 23(6):560-576, 2001.

[58] D. B. Reid. ”An Algorithm for Tracking Multiple Targets.” IEEE Transactions on Automatic Control, 24(6):843-854, 1979.

[59] R. L. Streit and T. E. Luginbuhl. ”Maximum Likelihood method for probabilistic multi-hypothesis tracking.” In proceedings of SPIE, volume 2235, 1994.

[60] C. Hue, J. Le Cadre, and P. Prez. ”Sequential Monte Carlo Methods for Multiple TargetTracking and Data Fusion.” IEEE Trans. on Signal Processing, 50(2):309-325, 2002.

[61] I.J. Cox and S.L. Hingorani. ”An effcient implementation of Reid’s multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 18(2):138-150, 1996.

[62] K.G. Murty. ”An algorithm for ranking all the assignments in order of increasing cost.” Operations research, 1968.

[63] T.J. Cham and J. M. Rehg. ”A Multiple Hypothesis Approach to Figure Tracking.” In IEEE International Conference on Computer Vision and Pattern Recognition, pp. 239-245, 1999.

[64] Y. Bar-Shalom and T.E. Foreman. Tracking and Data Association. Academic Press Inc., 1988.

[65] S. Birchfield. ”Elliptical Head Tracking Using Intensity Gradients and Color Histograms.” In IEEE Conf. on Computer Vision and Pattern Recognition, pp. 232-237, 1998.

[66] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. ”Robust Online Appearance Models for Visual Tracking.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 25(10):1296-1311, 2003.

[67] H. Tao, H.S. Sawhney, and R. Kumar. ”Object tracking with bayesian estimation of dynamic layer representations.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 24(1):75-89, 2002.

[68] M. Isard and J. MacCormick. ”BraMBLe: A Bayesian Multiple-Blob Tracker.” In IEEE Int. Conf. on Computer Vision, 2001.

[69] V. Vapnik. Statistical Learning Theory. Wiley NY., 1998.

[70] D. Terzopoulos and R. Szeliski. ”Tracking with Kalman Snakes.” In A. Blake and A. Yuille, editors, Active Vision. MIT Press, 1992.

[71] Michael Isard and Andrew Blake. ”CONDENSATION - conditional density propagation for visual tracking.” Int. Journal of Computer Vision, 29(1):5-28, 1998.

[72] J. MacCormick and A. Blake. ”Probabilistic exclusion and partitioned sampling for multiple object tracking.” Int. Journal of Computer Vision, 39(1), 2000.

[73] Y. Chen, Y. Rui, and T.S.Huang. ”JPDAF Based HMM for Real-Time Contour Tracking.” In IEEE Conf. on Computer Vision and Pattern Recognition, 2001.

[74] A. J. Viterbi. ”Error bounds for convolutional codes and an asymptotically optimum decoding algorithm.” IEEE Trans. Inform. Theory, 13:260-269, 1967.

[75] J.A. Sethian. Level set methods: evolving interfaces in geometry,fluid mechanics computer vision and material sciences. Cambridge University Press, 1999.

[76] M. Bertalmio, G. Sapiro, and G. Randall. ”Morphing active contours.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 22(7):733-737, 2000.

[77] A.R. Mansouri. ”Region tracking via level set PDEs without motion computation.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 24(7):947-961,2002.

[78] D. Mumford and J. Shah. ”Optimal approximations by piecewise smooth functions and variational problems.” Communication on Pure and applied Mathematics,42(5):677-685, 1989.

[79] Liang KH and Tjahjadi T Multiresolution Segmentation of Optical Flow Fields for Object Tracking. Applied Signal Processing, vol. 4 no. 3, 1997 (printed in 1998), 179-187

[80] D. Comaniciu and P. Meer. Mean shift analysis and applications.” In IEEE Int. Conf. on Computer Vision, volume 2, pp. 1197-1203, 1999.

[81] K. Toyama, B. Brumitt J. Krumm, and B. Meyers. Wallflower: Principles and Practices of Background Maintenance.” In IEEE Int. Conf. on Computer Vision, 1999.

[82] H. Chen, H. Lin, and T. Liu. Multi-object tracking using dynamical graph matching.” In IEEE Conf. on Computer Vision and Pattern Recognition, pp. 210-217, 2001.

[83] D. Cremers, T. Kohlberger, and C. Schnorr. Non-linear shape statistics in Mumford-Shah based segmentation.” In Europen Conf. on Computer Vision, 2002.

[84] J. Shi and C. Tomasi. Good features to track. Proc IEEE CVPR, pp. 593C600, 1994

[85] A. Elgammal, R. Duraiswami, D. Harwood and L. S. Davis Background and Foreground Modeling using Non-parametric Kernel Density Estimation for Visual Surveillance, Proceedings of the IEEE, July 2002.

[86] Lisa Gottesfeld Brown, “A Survey of Image Registration Techniques,” *ACM Computing Surveys*, vol. 24, no. 4, pp. 325-376, Dec. 1992.

[87] J. B. Antoine Maintz and Max A. Vergever, “A Survey of Medical Image Registration,” *Med Image Anal.*, vol. 2, no. 1, pp. 1-36, 1998.

[88] Josien P. W. Pluim, J. B. Antoine Maintz, and Max A. Viergever, “Multual-Information-Based Registration of Medical Images: A Survey,” *IEEE Transactions on Medical Imaging*, vol. 22, no. 8, pp. 986-1004, Aug. 2003.

[89] Barbara Zitová, Jan Flusser, “Image Registration Methods: A Survey,” *Image and Vision Computing*, vol. 21, pp. 977-1000, 2003.

[90] Paul Viola, William M. Wells III, “Alignment by Maximization of Mutual Information,” *International Conference on Computer Vision*, 1995.

[91] Paul Viola, William M. Wells III, “Alignment by Maximization of Mutual Information,” *International Journal on Computer Vision*, vol. 24, no. 2, pp. 137-154, 1997.

[92] Mert R. Sabuncu, Peter J. Ramadge “Spatial Information in Entropy-Based Image Registration,” *Biomedical Image Registration Lecture Notes in Computer Science*, pp. 132-141, 2003.

[93] Philippe Thévenaz, and Michael Unser, “Optimization of Mutual Information for Multiresolution Image Registration,” *IEEE Transactions on Image Processing*, vol. 9, no. 12, Dec. 2000.

[94] Josien P.W. Pluim, J.B. Antoine Maintz, and Max A. Viergever, "Image Registration by Maximization of Combined Mutual Information and Gradient Information," *IEEE Transactions on Medical Imaging*, vol. 19, no. 8, Dec. 2000.

[95] A. Collignon, F. Maes, D. Delaere, D. Vandermeulen, P. Suetens and G. Marchal, "Automated Multi-modality Image Registration Based on Information Theory," *Information Processing in Medical Imaging* 1995.

[96] Jeffrey Tsao, "Interpolation Artifacts in Multimodality Image Registration Based on Maximization of Mutual Information," *IEEE Trans. Med. Imag.* vol. 22, no. 7, pp. 854-864, 2003.

[97] Hong Chen, Ying-Qing Xu, Heung-Yeung Shum, Song Chun Zhu, Nan-Ning Zheng, "Example-Based Facial Sketch Generation with Non-parametric Sampling," *International Journal on Computer Vision*, 2001.

[98] Philippe Thévenaz, and Michael Unser, "A Pyramid Approach to Sub-Pixel Image Fusion Based on Mutual Information," *in Proc. IEEE Int. Conf. on Image Processing*, vol. 1, pp. 265-268, Sept. 1996.

[99] Yang-Ming Zhu "Likelihood Maximization Approach to Image Registration" *IEEE Transactions on Image Processing*, vol. 11, no. 12, pp. 1417-1426, Dec. 2002.

[100] Winston Li, Henry Leung "A Maximum Likelihood Approach for Image Registration Using Control Point and Intensity," *IEEE Transactions on Image Processing*, vol. 13, no. 8, pp. 1115-1127, Aug. 2002.

[101] Okello N., Ristic B. "Maximum Likelihood Registration for Multiple Dissimilar Sensor" *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 3, pp. 1074-1083, July 2003.

[102] Hong Chen, Ying-Qing Xu, Heung-Yeung Shum, Song Chun Zhu, Nan-Ning Zheng, “Example-Based Facial Sketch Generation with Non-parametric Sampling,” *International Journal on Computer Vision*, 2001.

[103] Huesman R.H., Klein G.J., Kimdon J.A., Kuo C., Majumdar S., “Deformable Registration of Multi-modal Data Including Rigid Structures,” *IEEE Symposium on Nuclear Science*, vol. 3, no. 10-16, pp. 1879-1882, Nov. 2002.

[104] Jan Kybic, Michael Unser “Fast Parametric Elastic Image Registration,” *IEEE Transactions on Image Processing*, vol. 2, no. 11, pp. 1427-1442, Nov. 2003.

[105] J.H. Mathews, *Numerical Methods for Mathematics, Science and Engineering*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1992.

[106] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes in C.* Cambridge, U.K.:Cambridge Univ. Press, 1992.